Phusion Passenger | iRobot Create | OpenFiler | AJAX | Squid

LINUX JOURNAL

Since 1994: The Original Magazine of the Linux Community
ISSUE 181 | www.linuxjournal.com

Turn an Old Computer into a Network Appliance with **OpenFiler**

Interview with **Neuros Technology's** Joe Born

COOL PROJECTS

Interact with the **iRobot Create**

Convert **8mm Film to DVD**

Develop an **Autonomous Underwater Vehicle**

Build a Linux-Powered **Rocket**

REVIEWED:

Home Automation with **Vera from Mi Casa Verde**

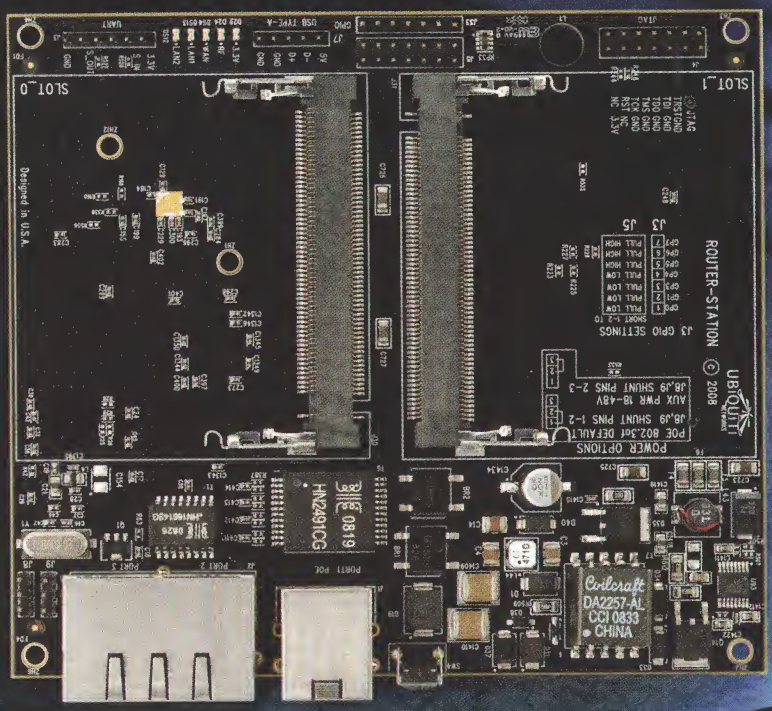
$5.99US $5.99CAN
05
0 09281 03102 4

UBIQUITI NETWORKS

The Embedded Wireless Dream Machine.

680MHz CPU, 64MB RAM, 16MB Flash, USB, 3 mini-PCI slots, 5A power supply for multiple hi-power radios, and enhanced temperature performance and ESD protection for carrier applications. Full Linux SDK support and a $59 USD MSRP.

RouterStation



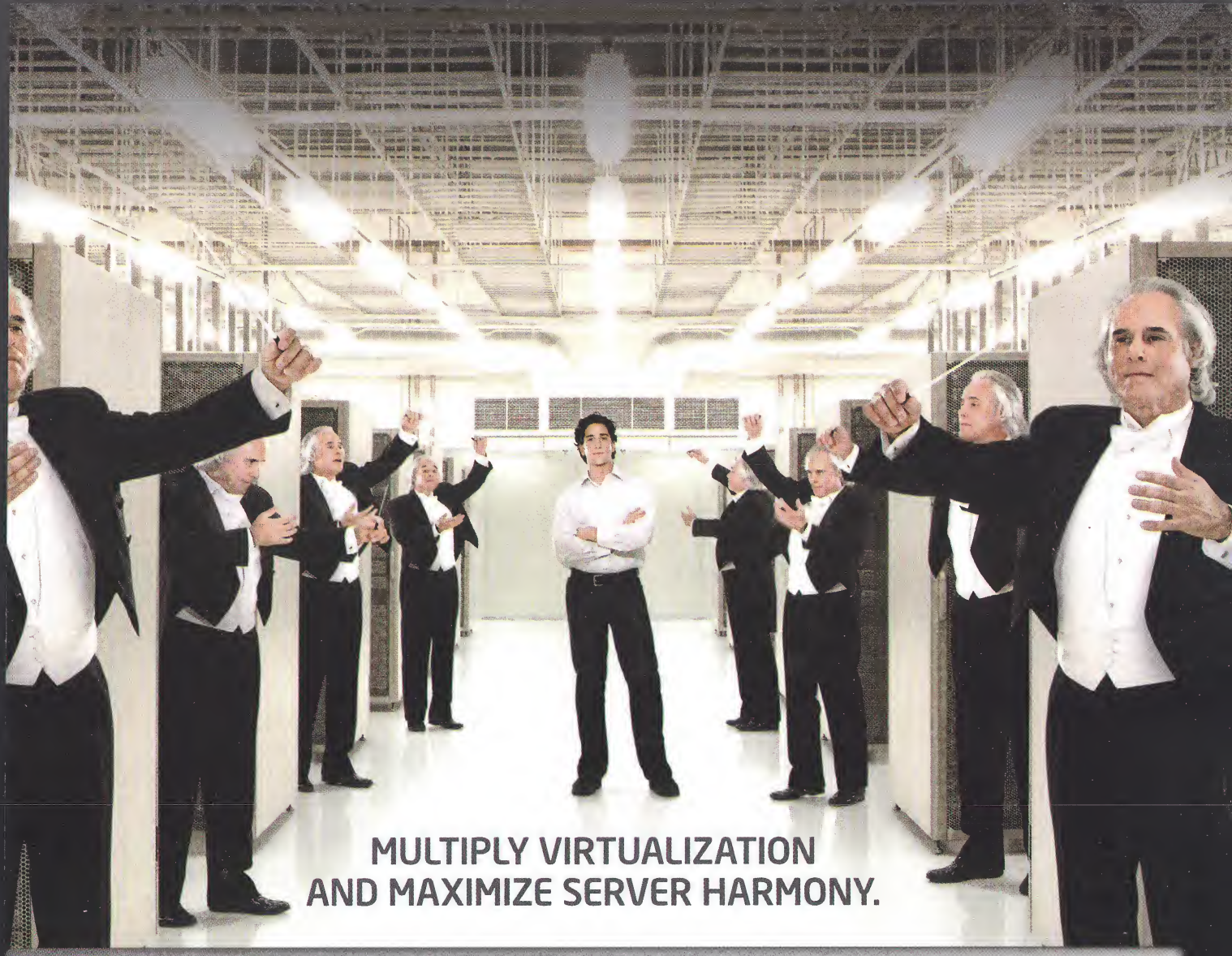
RouterStation UI CHALLENGE

CASH PRIZE
$200000
www.ubnt.com/challenge

www.ubnt.com

RouterStation UI Challenge: For Contest Details Visit: www.ubnt.com/challenge
Cash Prize $200000 (Two hundred Thousand) USD.

User Interface development based on OpenWrt Linux firmware. OpenWrt is a registered trademark of OpenWrt.

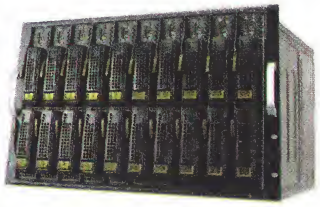


MULTIPLY VIRTUALIZATION AND MAXIMIZE SERVER HARMONY.

NIVEUS HTX WORKSTATION

- Intel® Xeon® processor
- Optional CPU overclocking and water cooling option
- Up to 32GB RAM
- BD / DVD-RW drive (Blu-Ray)

Call for Quote



BLADERUNNER II

- Intel® Xeon® processor
- 1600MHz front side bus
- 12MB level 2 cache
- Fully buffered DDR2-800, up to 32GB

Call for Quote

PENGUIN COMPUTING

www.penguincomputing.com
1-888-PENGUIN

Niveus HTX is the new flagship of Penguin's Personal Supercomputing and technical workstation line.
BladeRunner II is Penguin's new blade server platform, which combines high physical density with cutting edge Reliability, Availability, Serviceability (RAS) features.

Intel is not responsible for and has not verified any statements or computer system product-specific claims contained herein.

intel Xeon inside
Powerful. Efficient.

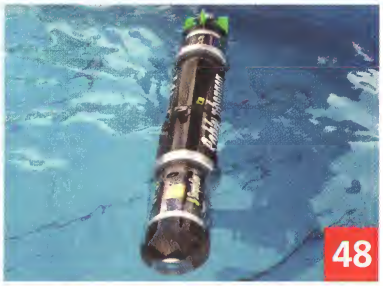
CONTENTS

MAY 2009
Issue 181

COOL PROJECTS

FEATURES

ON THE COVER



servers DIRECT®

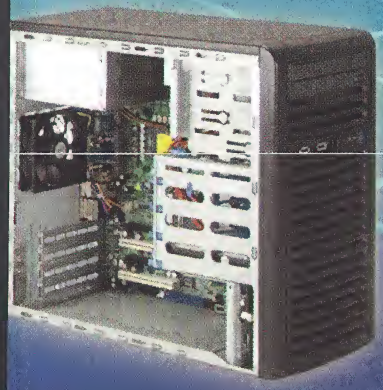
MORE PRODUCTS, BETTER SERVICE, GUARANTEED.

GO STRAIGHT TO THE SOURCE!

1.877.727.7887 | www.ServersDirect.com

YOUR HIGH PERFORMANCE COMPUTING HAS ARRIVED.

The ServersDirect® Systems with the Intel® Xeon® Processor helps you simplify computing operations, accelerate performance and accomplish more in less time.



SOHO SERVER / STORAGE

OPTIMIZED FOR SMALL OFFICE APPLICATIONS

SDP-S4100-T00

COST EFFECTIVE ENTRY LEVEL SERVER

- Mid Tower Chassis with 300W Power Supply
- Supermicro Server Board with Intel® 945GC Chipset
- 2x Realtek RTL8111C-GR Gigabit LAN Ports
- Support up to 4x SATA (3.0Gbps) Ports RAID 0, 1, 5, 10
- 4x 3.5" internal drive bays

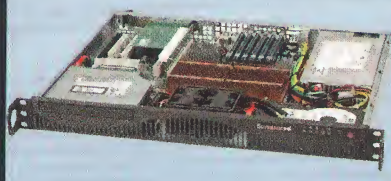
STARTING AT **$329**

SDP-S4200-T00

MOST COST EFFECTIVE LAN SOLUTION

- Mid Tower Chassis with 300W Power Supply
- Supermicro Server Board with Intel® 3200 + ICH9R Chipset
- Support up to a Dual-Core Intel® Xeon® 3000 Series processor
- Support up to 6xSATA (3.0Gbps) Ports RAID 0, 1, 5, 10
- 4x 3.5" internal drive bays

STARTING AT **$439**



SDR-S1208-T00

STARTING AT **$559**

- Supermicro Mini 1U Rackmount Server with 260W Power Supply
- Supermicro Server Board w/Intel® 946GZ Chipset
- Support up to a Dual-Core Intel® Xeon® 3000 Series processor
- TPM Support
- 1x 3.5" Internal Drive Bay
- 2x Intel® 82573 PCI-e Gigabit LAN Port

SDR-C1303-T02

STARTING AT **$899**

- 1U Rackmount Server with 400W Power Supply
- Supermicro Server Board w/ Intel® 5100 Chipset
- Support Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core
- Support up to 48GB 667/533MHZ DDR2 ECC Reg
- 2x 3.5" Hot-swap SATA Drive Bay
- Intel® 82573V & 82573L Gigabit Ethernet Controller

SDR-C2301-T06

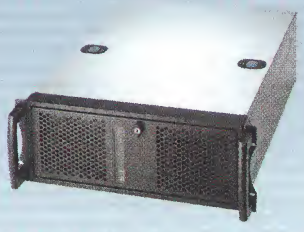
STARTING AT **$999**

- 2U Rackmount Server with 460W Power Supply
- Supermicro Server Board w/ Intel® 5100 Chipset
- Support Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core
- Support up to 48GB 667/533MHZ DDR2 ECC Reg
- 6x 3.5" Hot-swap SATA Drive Bay
- Intel® 82573V & 82573L Gigabit Ethernet Controller

SDR-C3301-T16

STARTING AT **$1,359**

- 3U Rackmount Server with 650W Power Supply
- Supermicro Server Board w/ Intel® 5100 Chipset
- Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core
- Support up to 48GB 667/533MHZ DDR2 ECC Reg
- 16 x 3.5" Hot-swap SATA Drives Trays
- Dual-port Gigabit Ethernet Controller



SDR-C4302-T02

STARTING AT **$899**

- 4U Rackmount Server with 600W Power Supply
- Supermicro Server Board w/ Intel® 5100 Chipset
- Support Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core
- Support up to 48GB 667/533MHZ DDR2 ECC Reg
- 3 x 3.5"Internal SATA Drives Trays
- Dual-port Gigabit Ethernet Controller

SDR-A8301-T42

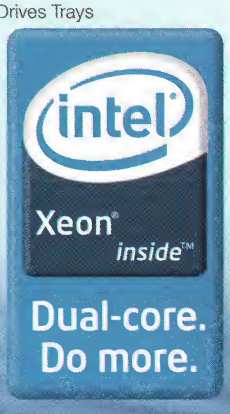
STARTING AT **$4,799**

- 8U Chassis with 1350W Redundant Power Supply
- Supermicro Server Board w/ Intel® 5400 Chipset
- Support Dual Intel® 64-bit Xeon® Quad-Core or Dual-Core
- Support up to 64GB SDRAM Fully Buffered DIMM (FB-DIMM)
- 42 x 3.5"Internal SATA Drives Trays
- Dual-port Gigabit Ethernet Controller

SERVERS DIRECT CAN HELP YOU CONFIGURE YOUR NEXT HIGH PERFORMANCE SERVER SYSTEM - CALL US TODAY!

Our flexible on-line products configurator allows you to source a custom solution, or call and our product experts are standing by to help you assemble systems that require a little extra. Servers Direct - your direct source for scalable, cost effective server solutions.

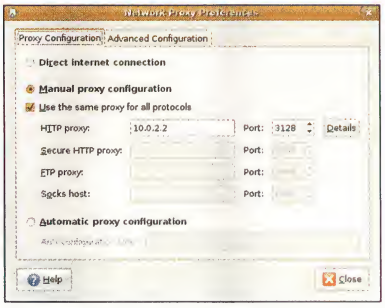
1.877.727.7887 | www.ServersDirect.com



CONTENTS

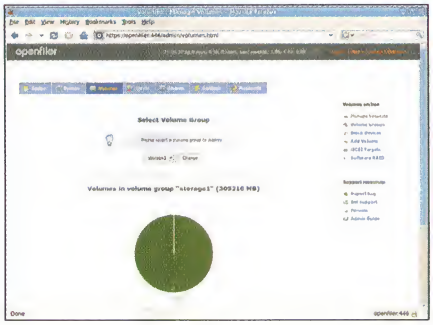
MAY 2009
Issue 181

COLUMNS



IN EVERY ISSUE

INDEPTH



REVIEW

70 JOE BORN, CEO OF NEUROS TECHNOLOGY

Next Month

READERS' CHOICE AWARDS 2009

Voting makes us feel good. With politicians the sensation is usually short-lived, but not so with the *Linux Journal* Readers' Choice Awards; it's a feeling that lasts all year long, and next month, we'll charge you up for another year. Find out who the winners are, and rest assured you picked 'em. There was no creative accounting, no Ponzi schemes, no Credit Default Swaps—nothing, just your votes.

Also in next month's issue, read how to build your own computing cluster with Rocks from Flintstone Computing (just kidding about the Flintstone part).

In Japanese, dojo means "place of the Tao". After reading how to create JavaScript grids with Dojo, you will know the true meaning of Tao Grasshopper.

USPS *LINUX JOURNAL* (ISSN 1075-3583) (USPS 12854) is published monthly by Belltown Media, Inc., 2211 Norfolk, Ste 514, Houston, TX 77098 USA. Periodicals postage paid at Houston, Texas and at additional mailing offices. Cover price is $5.99 US. Subscription rate is $29.50/year in the United States, $39.50 in Canada and Mexico, $69.50 elsewhere. POSTMASTER: Please send address changes to *Linux Journal*, PO Box 16476, North Hollywood, CA 91615. Subscriptions start with the next issue. Canada Post: Publications Mail Agreement #41549519. Canada Returns to be sent to Bleuchip International, P.O. Box 25542, London, ON N6C 6B2

zt Systems®

Dependable, Cost Effective Linux-Based Server Solutions

Designed for professional-grade performance, cost-effective ZT Reliant servers and desktops maximize your productivity and optimize your IT budget. An established USA-based manufacturer celebrating our 15th anniversary, ZT Systems combines individualized attention to your requirements with the experience and scalability you demand.

SC1107i-41

ZT Reliant SC1107i Small Business Server

Designed for Small Business Applications

- Intel® Core™ i7 Processor 920
- 6GB DDR3 1333MHz SDRAM (3x2GB)
- 1TB SATAII 32MB Cache Hard Drive (Max.4)
- 16x DVD-ROM
- nVidia GeForce 9400GT DDR2 DVI/VGA/HDMI
- RedHat Enterprise 5 Trial Version
- 3 Year Limited Warranty and 24x7 Lifetime Toll-Free Phone Support

Starting at only
$995.99*

4101Ci-84-C000001

ZT Reliant 4101Ci 4U Rack / Tower Server

Entry-level file/web/email Server Solution

- Single Intel® Xeon® Processor X3220
- 8GB DDR2 800 ECC Unbuffered SDRAM (4x2GB)
- (2) 1TB SATAII 32MB Cache Hard Drives
- 16x DVD-ROM
- 4 x 3.5" SAS/SATA Hotswapable Drive Bays
- RedHat Enterprise 5 Trial Version
- 3 Year Limited Warranty and 24x7 Lifetime Toll-Free Phone Support

Starting at only
$1095.99*

4203Ci-82-C000001

ZT Reliant 4203Ci 4U Rack / Tower Server

Flexible Server with enterprise-class SAS/SATA Support

- Dual Intel® Xeon® Processors X5405
- 16GB DDR2 ECC/REG. SDRAM (4x4GB, Max. 48GB)
- (3) 500GB SATAII 32MB Cache Hard Drives
- 16x DVD-ROM & 1.44MB Floppy Drive
- 4 x 3.5" SAS/SATA Hotswapable Drive Bays
- RedHat Enterprise 5 Trial Version
- 3 Year Limited Warranty and 24x7 Lifetime Toll-Free Phone Support

Starting at only
$1795.99*

intel Xeon inside™
Powerful. Efficient.

www.ztsystems.com/linuxjournal

866-984-7687

plus shipping and handling. System images may include equipment and/or accessories which are not standard features. Not responsible for errors in typography and/or photography.

LINUX JOURNAL

Executive Editor	Jill Franklin jill@linuxjournal.com
Senior Editor	Doc Searls doc@linuxjournal.com
Associate Editor	Shawn Powers shawn@linuxjournal.com
Associate Editor	Mitch Frazier mitch@linuxjournal.com
Art Director	Garrick Antikajian garrick@linuxjournal.com
Products Editor	James Gray newproducts@linuxjournal.com
Editor Emeritus	Don Marti dmarti@linuxjournal.com
Technical Editor	Michael Baxter mab@cruzio.com
Senior Columnist	Reuven Lerner reuven@lerner.co.il
Chef Français	Marcel Gagné mggagne@salmar.com
Security Editor	Mick Bauer mick@visi.com
Hack Editor	Kyle Rankin lj@greenfly.net

Contributing Editors
David A. Bandel • Ibrahim Haddad • Robert Love • Zack Brown • Dave Phillips • Marco Fioretti
Ludovic Marcotte • Paul Barry • Paul McKenney • Dave Taylor • Dirk Elmendorf

Proofreader	Geri Gale

Publisher	Carlie Fairchild publisher@linuxjournal.com
General Manager	Rebecca Cassity rebecca@linuxjournal.com
Sales Manager	Joseph Krack joseph@linuxjournal.com
Sales and Marketing Coordinator	Tracy Manford tracy@linuxjournal.com
Associate Publisher	Mark Irgang mark@linuxjournal.com
Webmistress	Katherine Druckman webmistress@linuxjournal.com
Accountant	Candy Beauchamp acct@linuxjournal.com

Linux Journal **is published by, and is a registered trade name of, Belltown Media, Inc.**
PO Box 980985, Houston, TX 77098 USA

Advertising
E-MAIL: ads@linuxjournal.com
URL: www.linuxjournal.com/advertising
PHONE: +1 713-344-1956 ext. 2

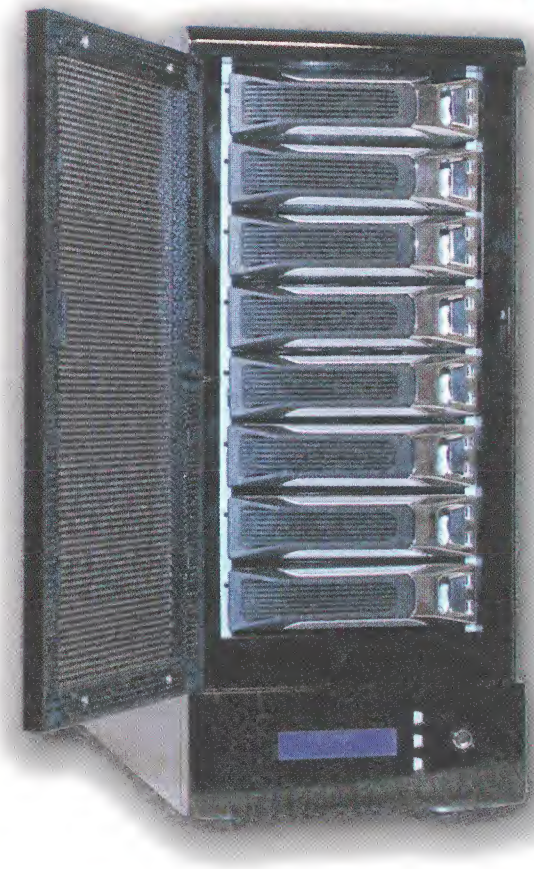
Subscriptions
E-MAIL: subs@linuxjournal.com
URL: www.linuxjournal.com/subscribe
PHONE: +1 818-487-2089
FAX: +1 818-487-4550
TOLL-FREE: 1-888-66-LINUX
MAIL: PO Box 16476, North Hollywood, CA 91615-9911 USA
Please allow 4–6 weeks for processing address changes and orders
PRINTED IN USA

Polywell Linux Solutions

More Choices, Excellent Service, Great Value!

Serving the Industry for More Than 20 Years

Netdisk 8000V
Quiet Performance NAS Storage



4TB $1,399
8TB $2,399
12TB $2,999

- Dual Gigabit LAN
- RAID-5, 6, 0, 1, 10
- Hot Swap, Hot Spare
- Linux, Windows, Mac
- E-mail Notification
- Tower or Rackmount

NVIDIA nFORCE

NVIDIA GeFORCE

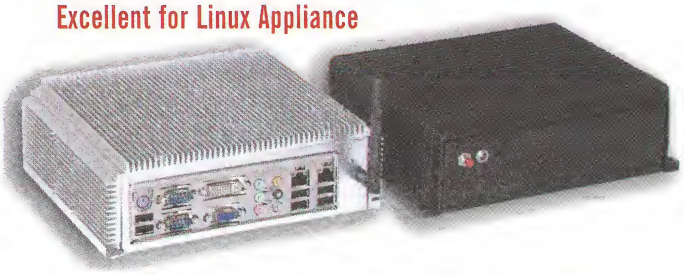
Silent Eco Green PC

NVIDIA® nForce Chipset and GeForce Graphics
Energy efficient, Quiet and Low Voltage Platform. starts at **$199**

The Best Terminal PC

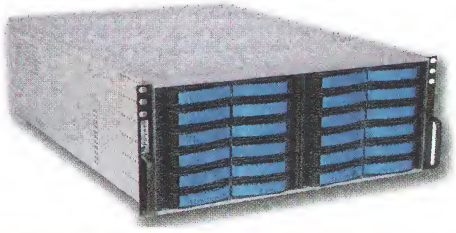
Fanless Silent ITX PC

1G DDR2, Solid State Drive starts at **$299**
Low-Voltage processor, Low-profile Add-on Available
Excellent for Linux Appliance



4U 24Bay 36TB Storage Server

Hardware RAID-6, NAS/iSCSI/SAN Storage
Mix SAS and SATA, 4 x GigaLAN or 10Gbit LAN



Mini-1U Server for Data Center ISP

Dual-Core or Quad-Core Processor
4GB to 8GB RAM, 2 x 500GB RAID HD
Linux Server Starts at **$499**

Polywell OEM Services, Your Virtual Manufacturer
Prototype Development with Linux/FreeBSD Support
Small Scale to Mass Production Manufacturing
Fulfillment, Shipping and RMA Repairs

- 20 Years of Customer Satisfaction
- 5-Year Warranty, Industry's Longest
- First Class Customer Service

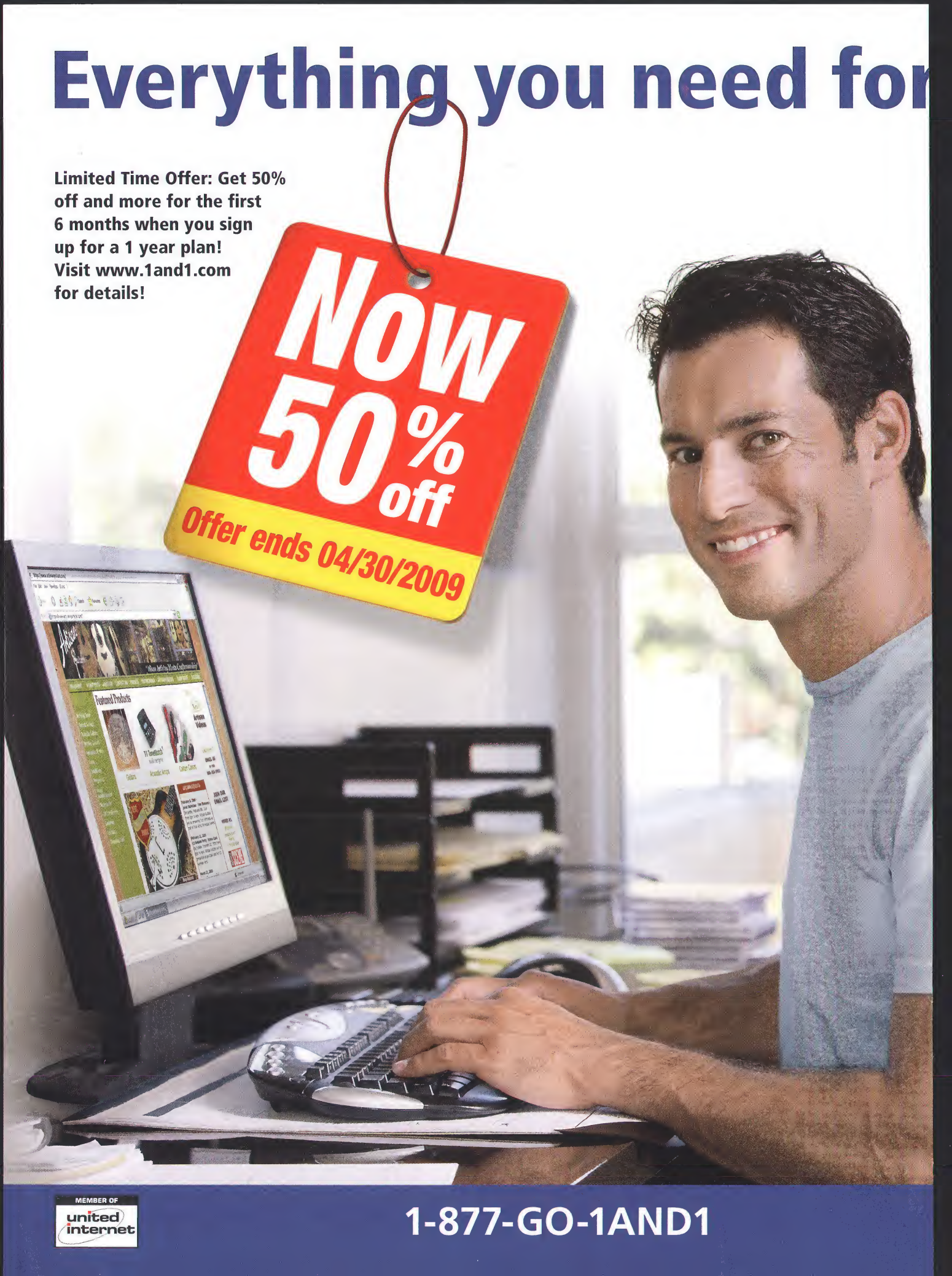
888.765.9686
linuxsales@polywell.com
www.polywell.com/us/Lx

POLY POLYWELL

Polywell Computers, Inc 1461 San Mateo Ave. South San Francisco, CA 94080 650.583.7222 Fax: 650.583.1974

Everything you need for

Limited Time Offer: Get 50% off and more for the first 6 months when you sign up for a 1 year plan! Visit www.1and1.com for details!



MEMBER OF united internet

1-877-GO-1AND1

a successful website

DOMAIN NAMES

.biz

Register your business website and save! Protect your personal information – private domain registration is included for FREE!

.biz domains ~~$8.99~~/year now $3.99/first year*

NOW 50% off
Offer ends 04/30/2009

WEB HOSTING

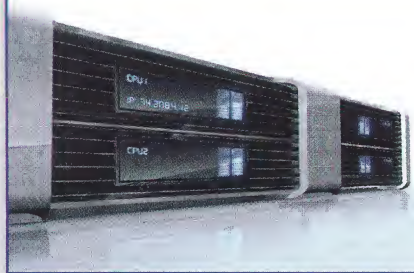
FREE Domain Names!

The 1&1 Business Package gives you everything you need for a professional website. 3 domains, site-building tool, 2,500 e-mail accounts, search advertising vouchers and more!

Starting at ~~$9.99~~/month now $4.99/mo.* (first 6 months)

NOW 50% off
Offer ends 04/30/2009

PREMIUM SERVERS



Designed specifically for high performance needs, these top-of-the-line AMD™ processors feature energy efficient technology, reducing costs and environmental impact with increased performance-per-watt. 1&1 matches 100% of the energy consumed in our data center with Renewable Energy Certificates.

Starting at ~~$199.99~~/month now $99.99/mo.* (first 6 months)

NOW 50% off
Offer ends 04/30/2009

E-COMMERCE SOLUTIONS

Set up your online store and start selling. Integration with eBay®, Shopzilla™, Google Product Search™ and Shopping.com® included!

Starting at ~~$24.99~~/month now $11.99/mo.* (first 6 months)

NOW 50% off
Offer ends 04/30/2009

Now accepting PayPal™

THIS COMPANY USES CERTIFIED CLEAN ENERGY

*Offers valid through April 30, 2009. Setup fee, minimum contract term, and other terms and conditions may apply. Visit www.1and1.com for full promotional offer details. Program and pricing specifications, availability and prices subject to change without notice.1&1 and the 1&1 logo are trademarks of 1&1 Internet AG, all other trademarks are the property of their respective owners. © 2009 1&1 Internet, Inc. All rights reserved.

1&1

www.1and1.com

SHAWN POWERS

Rockets and Robots and AJAX, Oh My

I won't even pretend the Cool Projects issue isn't my favorite of the year. This year is particularly neat for me, because although I'm on the editorial staff, before seeing the layout, I wasn't sure what content was going into it. So my first impressions were very similar to what yours are about to be. And, I think you'll be pleasantly surprised.

Ever since I was a kid, my definition of "something cool" almost always has included a robot. That includes books, movies and even science projects. Zach Banks proves this is the Cool Projects issue by showing us how to interface an iRobot with Linux. The iRobot isn't exactly as advanced as a Terminator robot from SkyNet, but it's also less likely to kill you. I think that's a fair trade-off.

Frank Pirz shows us his creation for digitizing old 8mm videotapes. Sure, there are services out there that will convert the old reel-to-reel tapes for you, but they're terribly expensive, and you have little control over the end product. Building a converter yourself solves all those problems. Yes, it's cool. Yes, it's homemade. And yes, it runs Linux. With all that new digital footage around, wouldn't it be nice to have an open-source method to play it on your television? Again, you're in luck. James Gray interviews Neuros Technology's CEO this month. There's not a more "open" company when it comes to video recording and playback, so you'll want to hear what he has to say.

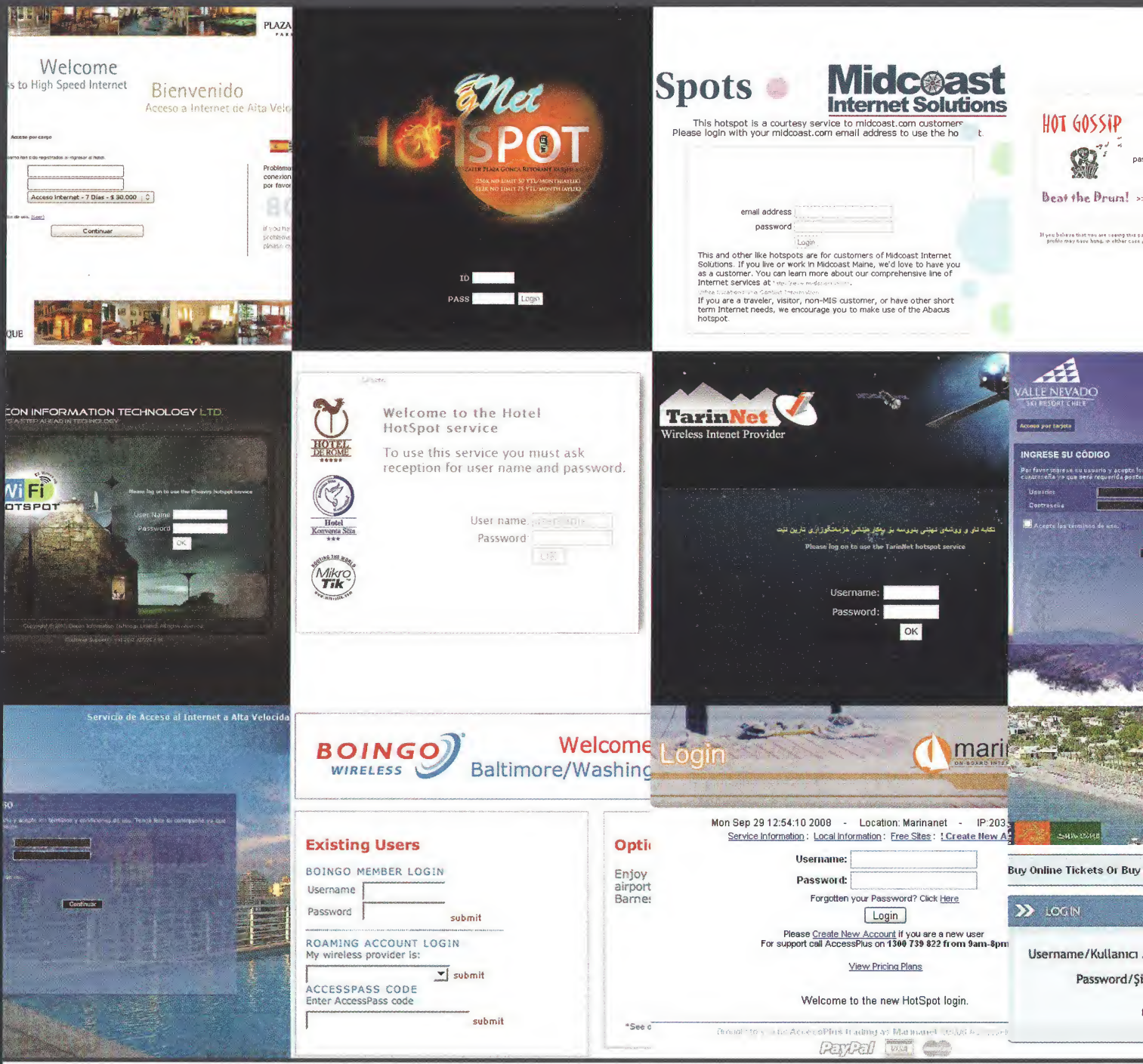
You'll also need a place to store all that video you digitize, and Bill Childers shows us an open-source storage appliance solution called OpenFiler. Many of the devices you can buy already are running Linux of some sort, so why not build your own with an old PC you have lying around? Repurposing old hardware to act as a file server is always cool, so be sure to check it out. And while you're at it, check out what Bill and Kyle are arguing about this month. Bill thinks AJAX is a great way to interface people with applications, but Kyle seems to think AJAX is more useful as a toilet cleaner. You be the judge of who makes the better argument.

Speaking of arguments, if you're like me, you have a hard time remembering to shut off the lights when you leave a room. In my house, this causes arguments that rival Bill and Kyle. Thankfully, Daniel Bartholomew shows us Vera, a home-automation device that can save you time and money—with Linux. It might be a gadget I can convince my wife to buy.

Not cool enough? Wow, tough crowd. Okay, we'll pull out the big guns. Well, maybe not actual guns, but the Cambridge Autonomous Underwater Vehicle sort of looks like a torpedo. Andy Pritchard tells us all about it. And, even bigger than that—rockets. No, really. Sarah Sharp shows us a rocket with a USB interface. If you think a USB interface means it's a tiny rocket, you'll be surprised. Be sure to look for pictures, because the scale will surprise you.

On the off chance your personality isn't similar to mine, fear not. This issue is focused on cool projects, but the coolest project of all is Linux. And, that is the focus every issue. This month, Mick Bauer continues his security series on Squid. Kyle Rankin shows us that even `rm -rf` can't keep a sysadmin down, and Reuven Lerner demonstrates running Rails applications with Phusion Passenger. Add Dave Taylor's article on special variables and Doc Searls' EOF article on Privacy, and this issue will keep you in Linux bliss all month. Unless SkyNet really does send killer robots from the future, in which case, this issue might actually save your life.■

Shawn Powers is the Associate Editor for *Linux Journal*. He's also the Gadget Guy for LinuxJournal.com, and he has an interesting collection of vintage Garfield coffee mugs. Don't let his silly hairdo fool you, he's a pretty ordinary guy and can be reached via e-mail at shawn@linuxjournal.com. Or, swing by the #linuxjournal IRC channel on Freenode.net.



MikroTik powered Hotspots around the world

MikroTik RouterOS powered hotspots are everywhere. From Internet cafes in desert towns of Africa, to futuristic airport lounges in the US and five star hotels in the Mediterranean. Mikrotik can power your systems too. Free evaluation installations are available in our download section.

www.mikrotik.com

letters

Windows Market Share

I tripped across Shawn Powers' video titled "An Open Video to HP" on YouTube [**www.linuxjournal.com/video/open-letter-hp**], and it occurred to me that the market share that Windows enjoys is actually very misleading in that there are a lot of Linux people who buy machines that come pre-installed with Windows and then toss out the Windows. That's what I did, and I know of others. So my point is (and I'm sure you probably thought of this already) that the Windows market share may not be as big as companies like HP are being led to believe. It would be nice if companies could be forced by law to sell machines without a pre-installed OS anywhere they market their machines.

--

Steve

I think you have a very good point. Sadly, I think OEM manufacturers get a significant kickback from the "crapware" they pre-install with Windows. My guess is that offsets the price of Windows for the OEM manufacturers, so they have little motivation to sell them without Windows. You're absolutely correct though; I have many computers with Windows license stickers on them that are running Linux. The numbers are probably skewed greatly regarding the installed base.—Ed.

sudo?

In his December 2008 article "Samba Security, Part II", Mick Bauer wrestles uneasily with sudo: "Note the sudo, necessary for Ubuntu. On other distributions, su to root...and omit the sudo that [begins each line]...." I've seen similar laments in other forums.

On systems like Ubuntu and Mac OS X, to avoid typing exhaustion and disruption to normal trains of thought, I "su to root" with:

```
sudo su
```

I haven't read *Linux Journal* for a while. Perhaps I'm missing something.

--

Henry Grebler

Mick Bauer replies: *If my writing style was awkward in this case, I apologize, but in fact, I'm quite comfortable with Ubuntu's requiring* `sudo` *for privileged commands. Habitually using root shells (including, I'm afraid, via* `sudo su`*) is a good way to make mistakes with an avoidably severe impact.*

The inconvenience of having to precede individual commands with `sudo` *is significantly offset by the fact that if you issue several in a row within a short period of time, you'll be prompted for your password only after the first command in the sequence.*

So again, I'd be the last to "lament" about this. On the contrary, I think the Ubuntu team has made a very sensible design choice with its sudo policy!

Be Distro-Neutral

When is *Linux Journal* going to change its name to *Ubuntu Journal*? For about two years now, I've seen a gradual migration from covering Linux in general to covering Ubuntu specifically. It's all well and good that most, if not all, of your writers use Ubuntu, but the rest of the community uses different distributions. I, for one, use OpenSUSE and have for well over five years. In fact, according to distrowatch.org, the second largest distribution in terms of "registered users" is OpenSUSE, and yet most of the mention I've been able to find regarding it feels like an afterthought.

I have no interest in switching to Ubuntu, Debian or any such distro. Why then do I have to feel like a secondary target in any article I read within *Linux Journal*? Worse yet, there are sidebars that seem to ignore completely the fact that other distros exist (see Mick Bauer's sidebar about regenerating the smb.conf file in Ubuntu/Debian in the December 2008 issue).

Perhaps it is time to find another source of Linux information—one that pertains to Linux in general and not what one magazine thinks I should be using.

--

Mathew Snyder

I understand your frustration. One of the difficulties with producing content that is beneficial to most people is that the procedures vary so widely from distribution to distribution. I'm guilty of using Ubuntu as an example often too. Sure, part of it is because it's the most popular distribution right now, but for me, it's also the one with which I'm most familiar.

We have had discussions internally about trying to make our content as distro-neutral as possible, so perhaps you'll see at least a slight shift in future issues. At least one of our staff members is a die-hard OpenSUSE fan, so you're certainly not alone. Thanks for the comment; it's important to be reminded of such things.—Ed.

Penguin in Your Pocket?

I could not believe my eyes when I received my [February 2009] copy of *Linux Journal* and caught sight of the cover. I wanted to ask it, "Is that a penguin in your pocket or are you really happy to see me?" Going for a different demographic? I am not insulted, but I almost choked on my coffee I was laughing so hard!

--

Charles Michaels

Bill Childers replies: *They say the camera can add ten pounds. Well, just like in First Life, cameras in* Second Life *can make objects appear larger than they are.*

Compression Tips

As usual, Mick Bauer's article, "Secured Remote Desktop/Application Sessions" in the September 2008 issue was overall excellent. If only I could have read it about three years ago, it would've saved me a lot of time researching all this stuff myself.

I noticed only one important detail that wasn't addressed. When using a graphical environment provided by a distant Linux or UNIX box, one frequently has performance issues, as the X window protocol isn't very compact. RFB is a lot better, but there's still a *lot* of data to transfer, and it's not compressed.

Of course, because it's all not compressed, there's a fairly simple solution: tell the ssh process we're tunneling through to compress the data stream, by giving it a -C command-line argument. This may not be needed when remotely administering your home Linux box from your laptop, hard-wired to your home gigabit Ethernet or even when using your 802.11n wireless network. When you're in the US and your server is in Australia (yes, I've done this), or even if you're just managing a server on the opposite coast of the US, the cost of compressing and uncompressing your data packets is going to be a lot less than the cost of getting the uncompressed data across that pipe.

For the advanced user, one can modify the gzip compression level using the GZIP environment variable. In my experience, -9 works best on very fast machines and intercontinental packets (when I was managing that GUI-only application in Australia, the difference between -8 and -9 actually was noticeable). On the other hand, unless you have a really slow link, when talking to the data center in the same building you're in, you will probably get the best speed from -1, if compression is even a net win.

Thanks also for your recent articles on Samba security [see Mick's Paranoid Penguin column in the November 2008, December 2008, January 2009 and February 2009 issues for the Samba articles]. About four months ago, my wife's boss gave her a Windows box for home use. As a result, I had a sudden interest in offering some Windows services from my home Linux server, and your series was very timely.

--
Ed

Mick replies: *Thanks so much for your kind words and your important compression tips! You're right, I completely overlooked the possibility of needing compression, which is so easily achieved with SSH and GZIP.*

Recovery Tip

In the article, "When Disaster Strikes: Hard Drive Crashes" [March 2009], Kyle Rankin advises as last resort when fsck can't get your files back to use strings to find your text data. Before doing that, I would suggest you try the great photorec tool (**www.cgsecurity.org/wiki/PhotoRec**). It originally was written to get photos back from dead Flash cards by looking for JPEG headers, but it now can identify hundreds of different file types on various filesystems.

--
Pascal Terjan

Kyle Rankin replies: *Thanks for the tip!*

PDF Slicing Tip

Regarding the "Slice and Dice PDF" Tech Tip in the February 2009 issue of *LJ* [p.40], I would like to point out that PDF slicing and more can be done using pdftk, without converting to PS and back to PDF. To do the same operation as the example in the tech tip, you need to issue the command:

```
pdftk afile.pdf cat 11-14 output file-p11-14.pdf
```

I think this is a little easier.

--
Stefano Canepa

Do It for the Goats

I've been an *LJ* reader on and off since 1996. I've had my current subscription for the past few years now, and I'm noticing with dismay the steady decline in technical articles on Linux internals. My favourite column used to be Kernel Korner. My current favourite is, perhaps unsurprisingly, the woefully short "diff -u". As tracking Linux core development is becoming more of a full-time job, those of us who can't afford the requisite time investment have to rely ever more on sources like *LJ* to avoid reaching the point where our systems are big black boxes to which we sacrifice the occasional goat in the hope that it'll appease the binary powers that be. For the sake of all those goats, would you consider carrying more articles akin to the LWN's "Kernel Development" section (currently my only reliable source of good technical Linux news)? It's not that I don't think browser comparisons, reviews of the latest desktops' new features and so on are a waste of ink, just that the information is more available elsewhere on-line for those who seek it, whereas with core Linux topics, not so much. I'm asking for a more balanced magazine, equally suited to the new multimedia-savvy, Web 2.0-type users who don't know (or care) what a bootloader is, as it is to the vim + gcc + xterm users who don't know (or care) how to access Twitter's newest features using the foo API. I realise this is generally easier said than done.

Thanks, and much respect for your dedication to the cause for all these years!

--
nessim

Thanks for your letter. It's a constant challenge to balance between articles that appeal to our super-techie crowd, and those that benefit the more desktop-oriented users. Because Linux is really beginning to show itself in less niche environments (Netbooks, mobile devices and so on), we do need to make sure those folks feel Linux Journal is for them too. That said, we'll make sure our hard-core geeks don't get left behind.

You'll probably see some variance between issues depending on the focus for that month, but we'll keep trying to balance our content so it appeals to our entire readership. Be sure to check out our upcoming Kernel Capers issue (August 2009).—Ed.

Simplifying Scripts

Regarding Dave Taylor's "Counting Words and Letters" article in the March 2009 issue: there are some options to tr that can be used to simplify Dave's script:

```
cat ^txt | tr '[:upper:]' '[:lower:]' | tr -cs
➥'[:alpha:]' '\n' | sort | uniq -c | sort -nr | head
```

tr accepts the '\n' argument. Also, the complement and squeeze options replace two calls to tr and one to grep. Plus, note that this eliminates counting spaces, which erroneously shows up as the second most-popular word in Dave's script.

--
Bruce Barnett

PHOTO OF THE MONTH

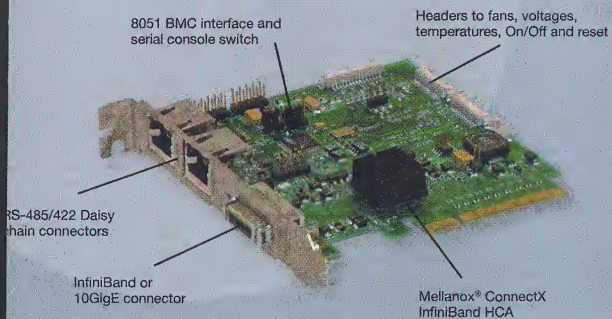
Have a photo you'd like to share with *LJ* readers? Send your submission to publisher@linuxjournal.com. If we run yours in the magazine, we'll send you a free T-shirt.

Richard Stallman and Chris Meloche from credil.org in front of the GNU Linux mobile, taken in Old Chelsea, Québec, Canada on January 26, 2009.

Your Applications Will Run Faster With Next Generation Microway Solutions!

TriCom™X

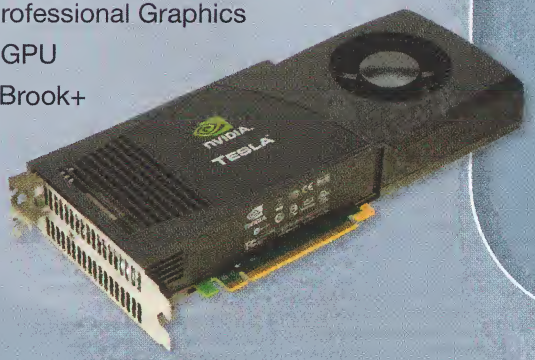
- QDR/DDR InfiniBand HCA
- ConnectX™ Technology
- 1µsec Latency
- Switchless Serial Console
- NodeWatch™ Remote Management

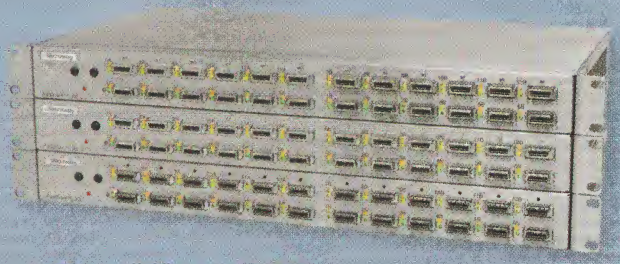


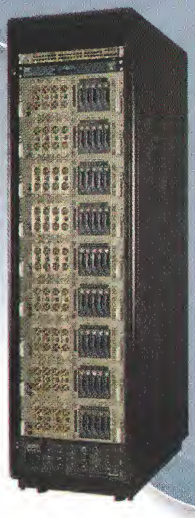
Teraflop GPU Computing

For Workstations and HPC Clusters

- NVIDIA® Tesla™ GPU with 240 Cores on One Chip
  - CUDA™ SDK
- NVIDIA® Quadro® Professional Graphics
- AMD® FireStream™ GPU
  - Stream SDK with Brook+







NumberSmasher®

Large Memory Scalable SMP Server

- Scales to 1 TB of Virtual Shared Memory
- Up to 128 CPU Cores
- 8U System Includes 32 Quad Core CPUs
- QDR 1 µsec Backplane

FasTree™X

- Mellanox® InfiniScale™ IV Technology
- QDR/DDR InfiniBand Switches
- Modular Design
- 4 GB/sec Bandwidth per Port
- QSFP Interconnects
- InfiniScope™ Real Time Diagnostics

Call the HPC Experts at Microway to Design Your Next High-Reliability Linux Cluster or InfiniBand Fabric.

508-746-7341

Sign up for Microway's Newsletter at
www.microway.com

Microway®
Technology you can count on℠

diff -u

WHAT'S NEW IN KERNEL DEVELOPMENT

Adam Osuchowski was poking around in the deep dark places of the kernel and came upon some hard-coded assembly that used the **xadd instruction**. Because the **386 CPU** didn't implement an xadd instruction, Adam asked whether Linux still supported the 386. The xadd instruction turned out to be just a bug, but the incident sparked a discussion about which older systems were and were not supported under Linux.

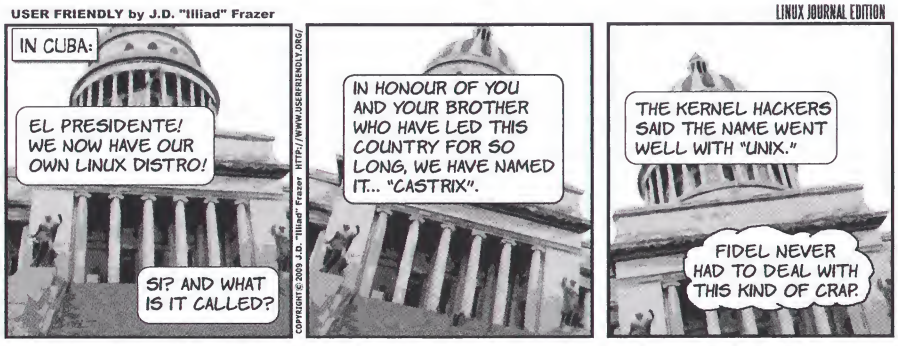
In terms of systems supporting **Symmetric Multi-Processing** (SMP), **Alan Cox** remarked that the first system to support **Intel's MP standard** was the **486 with external APIC**. He reckoned those would be the oldest systems capable of running SMP Linux, although he felt the earth may have been denuded of such systems long since. **Maciej W. Rozycki** commented:

> I failed to track down a single 486 SMP system that would adhere to the MP spec. There were, and possibly still are, APIC-based 486 SMP systems out there, but most likely they are not Intel MPS-compliant, by not providing the MP header at the very least. Thus, Linux would have to be ported, and I gather the interest in doing so is epsilon. Myself, I could not resist trying an APIC-based 486 SMP box and possibly fixing issues if I found one and it was MPS-compliant, but nothing beyond that I would say. Life's too short.

In terms of the 386, there was some speculation by various people, but no one could say for sure whether Linux would run on them. **Jan-Benedict Glaw** said he had an old, still functioning 386 that he'd dug out of storage, and that "it still powers on and boots up that ancient Debian version, using a 20GB (right, gigabytes) HDD." He said he might try experimenting with more current kernels and see whether they worked. Various other folks pointed out that 386 CPUs were still used in various embedded systems, and **Ingo Molnar** remarked that he knew of someone who occasionally booted up a 386 with current kernels.

So apparently the 386 is still kicking. My guess is the 286 is out of luck though—at least until someone decides to brave those strange waters.

—ZACK BROWN



HARD PLASTIC BOOKS THAT TALK

Last year at LinuxWorld, I had the opportunity to speak with Cliff Schmidt, the Executive Director at Literacy Bridge (**www.literacybridge.org**). At that point, Cliff was showing off an audio recording device with the eventual plan of being able to distribute sub-$10 gadgets that would allow for education and collaboration in struggling third-world countries. The little device that was literally in pieces back at LinuxWorld now is being used in Ghana as part of a pilot program.

Cliff Schmidt is the Executive Director of Literacy Bridge.

Although in many ways the less than $10 "Talking Books" lack features of the OLPC laptops, they also offer some advantages over their big brothers. The first is obviously in cost. Second, the audio-only interaction enables education where illiteracy often is a stumbling block. Paired with freely available audio recordings and the ability to record and share additional content, the Talking Books will be able to reach people that even the OLPC Project left behind.

—SHAWN POWERS

The Talking Books currently are being tested in Ghana.

LJ Index May 2009

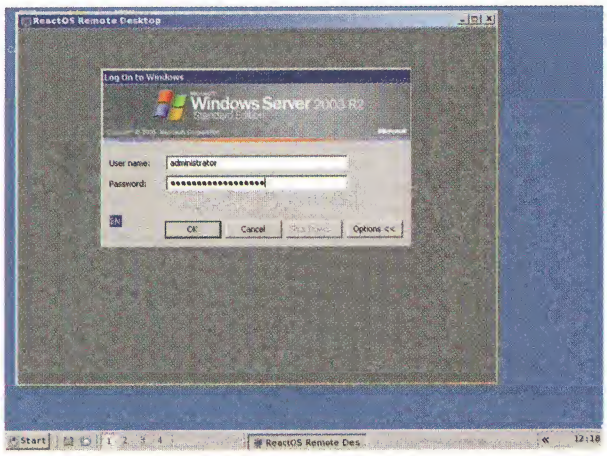
1. First issue to contain an *LJ* Index: **64**
2. Number of *LJ* Indexes in previous issues of *LJ*: **102**
3. Number of articles in previous issues of *LJ*: **4,338**
4. Google hits for "I Love Windows": **49,500**
5. Google hits for "I Hate Windows": **76,000**
6. Google hits for "I Love Linux": **75,300**
7. Google hits for "I Hate Linux": **5,660**
8. Google hits for "I Love Mac": **202,000**
9. Google hits for "I Hate Mac": **11,400**
10. Utility patent applications to the US Patent Office between 1790 and 2007: **13,154,369**
11. Utility patents issued (1790–2007): **7,301,128**
12. Utility patent approval rate (1790–2007): **55.5%**
13. Utility patent applications in 2007: **456,154**
14. Utility patents issued in 2007: **157,283**
15. Utility patent approval rate: **34.5%**
16. Best utility patent approval rate (1933): **86.2%**
17. Worst utility patent approval rate (1947): **26.7%**
18. Patent search results for the term "Linux": **7,810**
19. Patent search results for the term "Windows": **146,977**
20. Number of characters in Lincoln's Gettysburg Address: **1,476**
21. US National Debt as of 02/17/09, 1:30:44pm CST: **$10,776,246,598,791.76**

Sources: *1: grep | 2: grep | wc -l | 3: find | wc -l 4–9: Google | 10–19: patft.uspto.gov | 20. wc -l 21. www.brillig.com/debt_clock*

NON-LINUX FOSS

If you're a Linux fan, there's a bit of a tendency to think that Linux and open source are two ways of saying the same thing. However, plenty of FOSS projects exist that don't have anything to do with Linux, and plenty of projects originated on Linux that now are available on other systems.

Because a fair share of our readers also use one of those other operating systems, willingly or unwillingly, we thought we'd highlight here in the coming months some of the FOSS projects that fall into the above categories.



ReactOS Remote Desktop (from **www.reactos.org**)

We probably all know about our BSD brethren: FreeBSD, OpenBSD, NetBSD and so on, but how many of us know about ReactOS? ReactOS is an open-source replacement for Windows XP/2003. Don't confuse this with something like Wine, which allows you to run Windows programs on Linux. ReactOS is a full-up replacement for Windows XP/2003.

Assuming you consider that good news (a FOSS replacement for Windows), the bad news is that it's still only alpha software. However, the further good news is that it still is under active development; the most recent release at the time of this writing is 0.3.8, dated February 4, 2009. For more information, visit **www.reactos.org**.

—MITCH FRAZIER

Cool Projects Are Meant to Be Shared

This month's issue is all about cool projects, and we think the best part about making something cool is sharing it with the world.

Have you written some awesome software? Built a cool gadget? Taken something apart and repurposed its guts? We want to hear about it, and so do LinuxJournal.com readers.

The next time you have a cool project on your mind, whether it's complete or just a glimmer in your eye, log in to LinuxJournal.com and share it in our forums. Leave a comment on articles that inspire you, and let everyone know how you built a better mousetrap. Someone out there has topped Shawn Powers' DIY Arcade Game (**www.linuxjournal.com/article/9732**), right?

If you are short on time, try building yourself a virtual buddy with Chatbot::Eliza (**www.linuxjournal.com/content/it-live-or-it-chatboteliza**). Have fun, and don't forget to share your results at LinuxJournal.com!

—KATHERINE DRUCKMAN

Roku—Breaking the "Linux Not Invited" Rule

Many of you probably are familiar with the Roku media streaming device. In a partnership with Netflix, the Roku (**www.roku.com**) is one of several officially supported devices for streaming the large collection of Netflix's available movies and television shows. What makes the Roku interesting is that although Netflix doesn't support streaming its DRM-protected movies to Linux users, the Roku itself runs Linux.

The technology to stream Netflix titles to Linux is obviously available. Hopefully, as Linux users, we'll soon be able to join the Internet streaming club and watch movies on our desktops. Even more exciting will be media players like Boxee and XBMC (both of which run under Linux) being able to stream Netflix titles.

It is still frustrating that the streaming titles offered by Netflix are DRM-protected. The unmetered, on-demand streaming is a step in the right direction. Hopefully, in time, companies will realize that DRM only annoys those of us willing to spend money. It encourages pirating, rather than discouraging it.

—SHAWN POWERS

They Said It

Western society has accepted as unquestionable a technological imperative that is quite as arbitrary as the most primitive taboo: not merely the duty to foster invention and constantly to create technological novelties, but equally the duty to surrender to these novelties unconditionally, just because they are offered, without respect to their human consequences.

—Lewis Mumford

The drive toward complex technical achievement offers a clue to why the US is good at space gadgetry and bad at slum problems.

—John Kenneth Galbraith

The production of too many useful things results in too many useless people.

—Karl Marx

For a list of all the ways technology has failed to improve the quality of life, please press three.

—Alice Kahn

The real danger is not that computers will begin to think like men, but that men will begin to think like computers.

—Sydney J. Harris

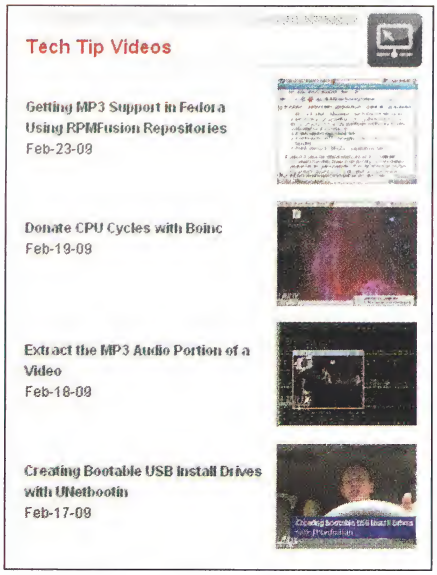
There is no subtler, no surer means of overturning the existing basis of society than to debauch the currency. The process engages all the hidden forces of economic law on the side of destruction, and does it in a manner which not one man in a million is able to diagnose.

—Vladimir Ilyich Lenin

Tech Tip Videos Now On-line

Get your daily how-to fix with LinuxJournal.com's weekly collection of Tech Tip videos. Each video is about one-minute long and walks you through cool tips and tricks. Check out the following:

- Getting MP3 Support in Fedora Using RPMFusion Repositories: **www.linuxjournal.com/video/getting-mp3-support-fedora-using-rpmfusion-repositories**
- Donating CPU Cycles with Boinc: **www.linuxjournal.com/video/donate-cpu-cycles-boinc**
- Extract the MP3 Audio Portion of a Video: **www.linuxjournal.com/video/extract-mp3-audio-portion-video**
- Creating Bootable USB Install Drives with UNetbootin: **www.linuxjournal.com/video/creating-bootable-usb-install-drives-unetbootin**



FREE TO A GOOD HOME: JUNK

I was pricing a low-end desktop computer the other day. When configuring it, I noticed that if I added a four-year warranty, it would cost more than the entire system! We've really come to the point where computer hardware is like a plastic fork. If a tine breaks off, it gets thrown away. Sadly, although throwing away plastic forks is rough on the environment, used computers are so much more so.

Thankfully, green is the new pink, and everyone seems to be interested in conservation and recycling. The problem is it's easier to talk about recycling computer hardware than to do it. I work at a school district, and we have a closet full of old CRT monitors just waiting for an opportunity to be recycled. There aren't any recycling places in our area, and thanks to the lead and glass, CRT monitors are very expensive to ship. So, they sit in a closet collecting dust.

Some amazing organizations out there are working hard to focus on another R, and rather than recycling old equipment, they reuse it. Places like Free Geek in Portland (**www.freegeek.org**), which I had the pleasure of touring last summer, take donated computer parts to create usable systems that are sold or donated back to the community. Thanks to Linux, those systems aren't encumbered with licensing issues. It's really a great way to get working, viable, stable computer systems in the hands of people who would likely never be able to afford one.

Although I'm not suggesting everyone should start a local Free Geek (although how cool would that be!), it's possible someone in your area already is doing something similar. Before you put that 17" CRT monitor and Pentium II computer on the curb, try giving it away in the local newspaper. If you like the idea of building computers for those in need, consider doing a small-scale version of Free Geek in your garage. Don't worry about running out of hardware, the local school district likely has computer parts piled in closets it would love for you to "recycle". With the power and flexibility of Linux, and the steady supply of aging computers, perhaps the path to world domination is by repurposing last year's Windows computers!

—SHAWN POWERS

Phusion Passenger

Run your Rails applications under Apache, using Phusion Passenger.

REUVEN M. LERNER

I've been using Ruby on Rails for several years now, and I continue to marvel at the ease with which I can create sophisticated Web applications. It's not perfect, but the fact is that Rails has made the hardest parts of Web development fairly painless. ActiveRecord, which lets me work with my database almost effortlessly, is obviously a great achievement, but the other elements of Rails—from database migrations to the templating system to the overall MVC structure—often surprise me with the elegant solutions they offer to common problems. The coming merger with Merb, a lean-and-mean alternative to Rails, leads me to believe that Rails will continue to provide developers with a terrific environment in which to practice their craft.

So, it's been frustrating to me, and to many other developers as well, that although Rails makes it easy to write applications, it makes the deployment of those same applications difficult. Sure, the famous screencasts in which you can create a blog make it clear that you can be up and running in almost no time. But, that's using WEBrick, a simple HTTP server written in Ruby, which no one realistically would use on a production site.

Apache, the HTTP server I have used since it was first released, and which continues to power the majority of Web sites in the world, would appear to be a natural choice for Rails deployment. After all,

So, it's been frustrating to me, and to many other developers as well, that although Rails makes it easy to write applications, it makes the deployment of those same applications difficult.

Rails is an open-source project, and just about every open-source Web framework hooks into Apache, right? Unfortunately not. The interface between Apache and Rails used a protocol known as FastCGI, or FCGI, and the combination of Rails, FCGI and Apache was long considered inferior to other options.

There always have been alternatives. Some sites used lighttpd, which had support for FCGI that was considered superior to what Apache offered. Others switched to Mongrel, which was designed in part to provide a stable and fast option for Rails applications. Some sites combined Mongrel with yet another open-source server, nginx (pronounced "engine-x"), which excels at handling static files. The book *Deploying Rails Applications*, which I recommend to anyone working on production Rails sites, steps through the configuration of Mongrel and nginx at great length.

For several years, then, deploying a Rails application meant learning to work with a new set of servers. This had several negative impacts. First, it raised the bar for using Rails just a bit more; now programmers needed to learn not only a new framework, but also a new HTTP server too. Another outgrowth was the relative dearth of hosting facilities that could work with Rails. PHP is nearly ubiquitous in the hosting world, in part because it integrates easily with the other elements of the LAMP stack (Linux, Apache and MySQL). Because Rails didn't easily integrate with Apache, it meant that hosting providers would need to learn a new skill and maintain a new package, which they weren't interested in doing.

And so, it was with a great deal of fanfare that Phusion, a Dutch consulting firm that has been using Ruby for the last few years, announced in 2008 that it had released Passenger, otherwise known as mod_rails, a module for Apache that makes it trivially easy to get up and running with a Rails application. I have switched to Passenger for my Rails production sites and have no complaints or regrets about doing so. And, it seems that I'm not alone; the company that originally sponsored the development of Ruby on Rails, 37signals, has indicated that it uses Passenger for some of its applications, and that it is thinking of moving additional applications to it in the future.

Yet another advantage to the fact that we can now use Apache to deploy Rails applications is the availability of other Apache modules. Apache was designed to be highly modular, letting developers include the modules they need, while excluding those that would make the server less efficient. Over the years, this has led to the development of dozens of different modules for Apache, covering everything from authentication to logging, from content negotiation to server administration. Having access to this large pool of useful modules means that our Rails application can be customized in a large number of different ways, providing us with

many choices when it comes to deployment.

This month, we look at how to use Passenger to deploy a Rails application. We also look at how we can combine other Apache modules with Passenger for a customized application solution.

Installation

Installing Passenger is a remarkably easy process, assuming that you already have Apache installed on your computer. First, you need to install the Passenger software, which comes as a Ruby gem:

```
sudo gem install passenger
```

This installs the Ruby gem (which on my Ubuntu server, is placed in /usr/lib/ruby/gems/1.8/gems), as well as several programs in /usr/bin, which we will use for Passenger. We use the first of these to install the Passenger module for Apache:

```
passenger-install-apache2-module
```

This starts the process of installing the Apache module on your computer; Passenger's installer script is smart enough to find many different versions of Apache, in many different places. It looks through Apache, determines what needs to be installed and then prompts you to install required packages automatically. For example, this is the output from the Passenger install program:

```
Checking for required software...

 * GNU C++ compiler... found at /usr/bin/g++
 * Ruby development headers... found
 * OpenSSL support for Ruby... found
 * RubyGems... found
 * Rake... found at /usr/bin/rake
 * Apache 2... found at /usr/sbin/apache2
 * Apache 2 development headers... not found
 * Apache Portable Runtime (APR) development headers... found
 * Apache Portable Runtime Utility (APR) development headers... found
 * fastthread... found
 * rack... found
```

If you are missing one or more of these programs, the installer tells you what commands you need to run in order to install the necessary programs. For example, my Ubuntu server indicated that I needed to install Apache 2 development headers and suggested I do this by executing the following:

```
apt-get install apache2-prefork-dev
```

I followed those instructions, and it worked. Once I finished installing the additional package via apt-get, I re-ran `passenger-install-apache2-module`. This time around, it succeeded, compiling the Apache module and adding an appropriate LoadModule directive in the Apache configuration file.

Indeed, now that Passenger is on our system, we can configure one or more Web sites. A simple configuration—indeed, the shortest one—would look like this:

```
<VirtualHost *:80>
    ServerName www.mysite.com
    DocumentRoot /home/reuven/public
</VirtualHost>
```

Note that the DocumentRoot points to the public directory of the Rails application, rather than to the Rails root. The Rails application itself is assumed to reside in the app directory parallel to public. Assuming that your Rails application is in place, restarting the Apache server will load the Passenger module, then run your application. By default, Passenger assumes you want to run your application using the "production" environment, which is optimized for system efficiency, rather than programmer interactivity. You can use the RailsEnv configuration directive to set the environment to something else, however:

```
RailsEnv development
```

Once your server is running, Apache continues to produce its standard log files (that is, error, access and referrer). Rails also will produce its standard log files in the application's log directory, so if you are used to looking through logs/production.log, you need not fear that it will be going away.

To restart the Rails application, you need to create a file called restart.txt in the application's tmp directory. Once this file is created, Passenger restarts the application, making sure not to interrupt any HTTP requests that it is currently servicing. (In this way alone, it is clearly superior to restarting Apache completely.)

Capistrano

If you use Capistrano to deploy your programs to one or more production servers, you might be wondering how it works with Passenger. The answer is that Capistrano works just fine, but you do need to consider the layout of a Capistrano-enabled server to ensure that everything works correctly.

As you might know, Capistrano keeps several versions of a Web application around. Each version is stored in its own directory, within the releases directory. A symbolic link, called current, points to the subdirectory inside of releases that corresponds

to the current version. This means that reverting to a previous version is nearly instantaneous, because it involves redefining the symlink to point to a previous subdirectory of releases.

So, on a Capistrano-enabled system, you will want your Apache configuration to look like the following:

```
DocumentRoot /home/reuven/current/public/
```

Notice the introduction of /current into the DocumentRoot. This tells Apache that it should use the current symbolic link and, thus, treat whatever current points to as the live version of the application.

But, what happens when you want to deploy a new version of your application? Capistrano is smart enough to rewrite the symbolic link, but it doesn't natively know how to restart the server. Fortunately, as we saw before, a restart involves creating the restart.txt file, so a Passenger-friendly recipe (inside of deploy.rb) could look like this:

```
namespace :deploy do
  desc "Restart Application"
  task :restart, :roles => :app do
    run "touch #{current_path}/tmp/restart.txt"
  end
end
```

Now, when we issue the `cap deploy` command, it knows to restart the server by creating restart.txt in the application's tmp directory. If we are interested only in restarting the server, we can do so by issuing the `cap deploy:restart` command, which runs just the restart task inside the deploy namespace.

Monitoring

Passenger comes with a number of utility programs that make it easy to keep track of your server's status and resource use. The program passenger-memory-status, for example, lists all the current processes being used by Apache, as well as the number of threads that each process has spawned. It then describes the amount of memory that each of those processes is using. For example, here is the memory usage report for ten Apache processes on a production Web server:

```
root@kipling:~# passenger-memory-stats
-------------- Apache processes --------------
PID    PPID   Threads  VMSize     Private  Name
----------------------------------------------
2941   15559  1        11.9 MB    0.5 MB   /usr/sbin/apache2 -k start
2944   15559  2        132.5 MB   9.1 MB   /usr/sbin/apache2 -k start
7392   20753  27       234.0 MB   6.8 MB   /usr/sbin/apache2 -k start
13383  20753  2        124.0 MB   7.9 MB   /usr/sbin/apache2 -k start
15559  1      1        11.9 MB    0.5 MB   /usr/sbin/apache2 -k start
15563  15559  2        147.7 MB   8.7 MB   /usr/sbin/apache2 -k start
17357  20753  1        11.9 MB    0.5 MB   /usr/sbin/apache2 -k start
17362  20753  27       239.8 MB   12.8 MB  /usr/sbin/apache2 -k start
17477  20753  27       236.6 MB   7.8 MB   /usr/sbin/apache2 -k start
20753  1      1        11.9 MB    0.4 MB   /usr/sbin/apache2 -k start
### Processes: 10
### Total private dirty RSS: 54.95 MB
```

That same command also shows us the current memory status for our Passenger (that is, Ruby) processes. It shouldn't come as any surprise to learn that the Ruby processes typically will be much larger than the Apache ones. Indeed, monitoring the memory usage of the Rails processes is an important thing for Rails developers to do; without such feedback, it will be difficult to measure how efficiently processes are working.

Other Apache Modules

Finally, as I mentioned previously, one of the best parts of using Apache for Rails applications is the fact that you can mix and match other Apache modules, as you like. For example, I am a big fan of both mod_status and mod_info, two modules for Apache that make it possible to peek into the server's current configuration and execution state.

In the same way, I wanted to compress files automatically as they were sent from my server to the user's browser. By incorporating mod_deflate into my server configuration, I was able to add automatic, on-the-fly compression with the following directive:

```
SetOutputFilter DEFLATE
```

Finally, I recently worked on a simple Rails site that wanted to restrict access to items under the /admin URL to authorized users. I could have used a Rails plugin, such as restful_authentication, but as I was using Passenger, I thought it might be just as easy and fast for me to use HTTP authentication on the site, defined in the Apache configuration file. Sure enough, the following was enough to do the trick:

```
<Location /admin>
  AuthName "Site admin"
  AuthType Basic
  AuthUserFile /opt/mysite/users

  require valid-user
</Location>
```

Of course, you could argue that this sort of authentication is far less flexible than a Rails-based

one, and you would be right. But for a site that has very simple needs, and that doesn't need something as fancy as restful_authentication, Apache's built-in (and well documented) HTTP authentication is a good solution.

Conclusion

The beauty of Apache is its flexibility, and Passenger makes it possible for us to incorporate that flexibility into our Rails applications, using the same server software that we've used for years.

Phusion Passenger has made it easier to deploy Rails applications, which is a good thing for Rails developers everywhere. It not only allows you to use your existing knowledge of the Apache server, but also means you can incorporate some of the many modules that have been developed for Apache over the years.■

Reuven M. Lerner, a longtime Web/database developer and consultant, is a PhD candidate in learning sciences at Northwestern University, studying on-line learning communities. He recently returned (with his wife and three children) to their home in Modi'in, Israel, after four years in the Chicago area.

Resources

You can learn more about Ruby on Rails at **www.rubyonrails.com**. Information about Phusion Passenger is at **www.modrails.com**. The site contains a great deal of documentation, including a full list of configuration directives that allows you to customize fully the way that Passenger is deployed for your site.

The book *Deploying Rails Applications*, published by the Pragmatic Programmers and written by several well-known Rails developers, doesn't include a description of Passenger. But, it does have a large number of other, good suggestions for rolling out Rails applications, and all Rails developers would do well to look at this book, including the many useful hints that it offers.

DAVE TAYLOR

More Special Variables

Use bash's more powerful variable substitution forms to simplify your scripts.

I realize this might throw a spanner into the editorial works here at *Linux Journal*, but after a two-month sidetrack on how to analyze letter usage in English to give you an edge in *Hangman* (yeah, I can't believe I write about this stuff either), it's time to get back to our tour of basic shell variable referencing capabilities.

In previous columns, we talked about ${var:-alt value}, ${var:=alt value}, ${var:?no value} and even ${var:start:length} as a way to extract specific ranges of characters from a variable.

This month, I want to look at what are perhaps some of the more arcane variable references you can do—calls that are definitely helpful if you're deep in the zone with your scripting. I imagine they won't be things you need for those quick five-line scripts, but when your little project has expanded to a dozen screens and you have seven functions and a dozen arrays, well, these will be of great value to you.

Expanding and Matching

In a previous column, I showed how to do substring expansion with shell variables in the form of ${var:start:length}, but it's also useful to know the length of a variable's value. This can be done with ${#var}, like this:

```
$ test="the rain in Spain"
$ echo ${#test}
17
```

One situation I've encountered in scripts is the need to set an arbitrary number of variables in the form value1, value2, value3 and so on. Later, I need to determine the names of the ones that I've set. My lazy solution is typically another variable, valuecount, which counts the number of variables I've set, but, of course, that doesn't directly give me the names. A smarter way to do this is with the ${!pattern*} notation, as shown here:

```
$ echo ${!t*}
test
$ thimble="full"
$ tart="pop"
$ echo ${!t*}
tart test thimble
```

As you can see, it lets you get a list of defined variables that match the specified pattern. I'm using t* in the example, but it just as easily could be value* to match the situation outlined earlier.

Pattern Substitution

Here's a cool thing you can do with the bash shell that I'm betting you didn't realize: pattern substitution. When I have a situation where this is required, I almost always use the clunky and CPU-expensive form of:

```
var=$(echo $var | sed 's/old/new')
```

which actually can be neatly accomplished with the shell itself by using the form ${var/old/new}. I kid you not! Check out this example:

```
$ test="The Rain in Spain"
$ echo ${test/ain/ixn}
The Rixn in Spain
```

If you're like me, your fingers are itching to add a /g suffix to the substitution. It turns out that's done a bit differently within a shell variable: you need to have the pattern start with a /, which looks a bit weird, but it does work:

```
$ echo ${test//ain/ixn}
The Rixn in Spixn
```

The general case here is ${var//pat/global subst}. There's more you can do with this notation too—notably, use the equivalent of the ^ and $ special characters you might use in sed regular expressions to root the pattern to the beginning or end of the variable's value:

```
$ echo ${test/#ain/ixn}
The Rain in Spain
$ echo ${test/%ain/ixn}
The Rain in Spixn
```

In the first situation, the pattern didn't match the first few letters of the variable value (the pattern would need to have been "The" rather than "ain"), so nothing changed. In the second situation, however, it did match the last few characters, so the substitution took place.

To be fair, using sed does give you quite a bit more power and capability, but if you're just doing something simple like removing an extension and appending a PID to a variable to make a quick temp file, you can indeed just use shell pattern replacement:

```
$ test="The Rain in Spain.txt"
$ echo ${test/%.*/}.$$
The Rain in Spain.10126
```

Personally, I think this is very cool!

Command Substitutions

We've explored just about everything you can do with variables other than delving into arrays, which we'll do next month, so I thought I'd take a bit of space to show you a few slick command substitution tricks. First off, us old-timers are used to using backticks to have a command embedded within another, as in the following:

```
echo the date is `date`
```

This is pretty commonly used, but, in fact, a better and certainly more readable notational convention is to use $() instead, as I showed earlier. This is functionally identical:

```
echo the date is $(date)
```

Using this notation gives you some interesting capabilities. For example, instead of `$(cat file)`, you simply can use `$(< file)` to make the contents of the file appear.

As is always the case with the shell, when and where fields are parsed is important too. Check out the following:

```
$ echo the date is $(date)
the date is Wed Feb 4 08:08:35 MST 2009
$ echo the date is "$(date)"
the date is Wed Feb  4 08:08:43 MST 2009
```

By adding the double quotes around the second invocation of $(date), you can see that the returning values weren't parsed by the shell and normalized: notice the two spaces between Feb and 4 in the second output compared to one space in the first output.

I hope I don't need to tell you what happens if you use single quotes instead of double quotes—oh, what the heck:

```
$ echo the date is '$(date)'
the date is $(date)
```

No surprise there—single quotes disable shell expansion, just as it does in this case:

```
$ echo The '$HOSTNAME' is $HOSTNAME
The $HOSTNAME is soyvah33
```

This leads to the classic question of what if you actually do want those quotes to be part of the output? It's a bit convoluted, but this works:

```
$ echo The '$HOSTNAME' is \'$HOSTNAME\'
The $HOSTNAME is 'soyvah33'
```

Let's wrap things up here, and next month, we'll dig into the oft-confusing world of shell script arrays.■

Dave Taylor has been involved with UNIX since he first logged in to the ARPAnet in 1980. That means, yes, he's coming up to the 30-year mark now. You can find him just about everywhere on-line, but start here: www.DaveTaylorOnline.com.

Building a Secure Squid Web Proxy, Part II

MICK BAUER

Get a Squid caching proxy up and running, securely.

Last month, I began a series of articles on Squid Web proxy security by introducing the theory, benefits and architecture of Web proxies. This month, we dive right in to basic Squid installation, configuration and testing, and begin hardening our Squid proxy.

What We're Doing (Review)

As you'll recall from last month, a Web proxy provides a control point for restricting which external Web sites your users can reach. It allows you to permit Web access without allowing non-Web traffic (or even publishing a default route to the Internet), and it provides a convenient place to perform content filtering and transaction logging.

As you also may recall, unlike a firewall, a Web proxy doesn't need to be a physical choke point through which all traffic must pass for a physical path to the outside. Instead, you can use firewall rules or router ACLs that allow only Web traffic, as a means of ensuring your users will use the proxy. Accordingly, your Web proxy can be set up like any other server, with a single network interface.

This is the case with the Web server I show you how to build in this and subsequent columns. This month, we focus on Squid itself; we'll cover add-ons like SquidGuard in future columns.

Obtaining and Installing Squid

So, where do you get Squid software? Naturally, the Squid Web site (see Resources) is the definitive source. But, because Squid has been the gold standard for Linux Web proxies for so many years, chances are it's a fully supported package in your Linux distribution of choice. If so, that's how I recommend getting it; it's easier to keep it patched that way, and you'll have greater assurance of stability and compatibility with the other things on your system.

On Ubuntu and other Debian variants (not to mention Debian itself), you need the packages squid and squid-common. On Red Hat and its variants, you need the package squid. And, on SUSE and OpenSUSE systems, you need squid.

At the time of this writing, all three of these families of distributions (Debian, Red Hat and SUSE) are maintaining separate packages for Squid version 3; the packages cited above are for version 2. This is because although the Squid development team recently declared Squid 3.0 to be a stable release (in November 2008), at the time of these three distributions' most recent production releases, Squid 3.0 still was considered to be a beta code branch, with 2.6 or 2.7 as the preferred production versions.

On the one hand, by the time you read this, Squid 3.0 (maybe even 3.1, which is in beta right now) may be mainstreamed into your Linux distribution of choice. On the other hand, maybe not. So for now, I'm going to use examples from Squid 2.6.18, the version on my Ubuntu system. They still should be perfectly valid for later versions—generally, later versions have *additional* options and features, not *replaced* options. I can cover Squid 3.0 in a future column.

I leave it to you to use the package manager of choice to install Squid packages on your RPM-based system, but on Debian-based systems, the most direct way is usually with the command:

```
bash-$ sudo apt-get install squid
```

(apt-get automatically will determine that it also needs squid-common and will install that too.)

By the way, you do *not* need to install Apache or any other Web server package on your Squid server, unless, of course, you're also going to use it as a Web server or want to use some Web-based administration tool or another. Squid itself does not need any external Web server software or libraries in order to proxy and cache Web connections.

Squid itself does not need any external Web server software or libraries in order to proxy and cache Web connections.

Configuring Squid: Basic Functionality

Creating a basic, working configuration for Squid isn't much harder than installing it. Like so much else in Linux, it's a matter of making small changes to a single text file, in this case, squid.conf. In all three distribution families I mentioned, its full path is /etc/squid/squid.conf.

To get started, first open a command window,

and back up the default squid.conf file (non-Ubuntu users can `su` to root and omit the `sudo` from these examples):

```
bash-$ cd /etc/squid
bash-$ sudo cp squid.conf squid.conf.default
```

Next, open squid.conf with your text editor of choice. You actually may prefer a graphical editor, such as gedit, but I've always used vi for its simplicity and ubiquity—if it's UNIX-like, it's got vi.

(Note to the emacs-loving alpha geeks among you: yes, emacs is more powerful; it's written in LISP; God kills a kitten every time someone installs Gvim; you win! But, I still like vi.)

Believe it or not, all you need to do to get Squid running is add two lines to the ACL (Access Control List) section of this file: an object definition that describes your local network and an ACL allowing members of this object to use your proxy. For my network, these lines look like this:

```
acl mick_network src 10.0.2.0/24
http_access allow mick_network
```

The first line is the object definition. The `acl` signifies that I'm about to define an ACL object. `mick_network` is the name I've chosen for this object. `src` means that it represents the IP address or range of addresses of hosts initiating TCP transactions with my proxy (that is, proxy clients). Finally, `10.0.2.0/24` is my LAN's network address in CIDR notation, which in this case translates to "the range of IP addresses from 10.0.2.1 through 10.0.2.254".

The second line declares an actual ACL: allow transactions involving the object mick_network—that is, transactions initiated by hosts having IP addresses in the range 10.0.2.1 through 10.0.2.254.

If more than one network address comprises your local network, you can specify them as a space-delimited list at the end of the acl statement, for example:

```
acl mick_network src 10.0.2.0/24
192.168.100.0/24
```

Because ACLs are parsed in the order in which they appear (going from top to bottom) in squid.conf, do *not* simply add these acl and http_access lines to the very end of squid.conf, which will put them after the default "http_access deny all" statement that ends the ACL portion of the default squid.conf file. On my Ubuntu system, this statement is on line 641, so I inserted my custom acl and http_access lines right above that.

In case you haven't guessed, `all` is a wild-card ACL object that means "all sources, all ports, all destinations" and so forth. Any transaction that is evaluated against any http_access statement containing `any` *will* match it, and in this case, will be dropped, unless, of course, it matches a preceding http_access line.

Now that you've created an object and ACL for your local network, you should save squid.conf and then restart Squid by typing this command (see earlier note about su root shells vs. sudo):

```
bash-$ sudo /etc/init.d/squid restart
```

In fact, if you're editing squid.conf from a `sudo vi squid.conf` session, you don't even need to leave your editing session; just do a `:w` to save your work, then type `:! /etc/init.d/squid restart` to restart Squid from within vi.

To test whether things are working, you need to configure a machine other than the proxy itself to use your proxy. (Squid comes configured by default to allow transactions from 127.0.0.1, the local loopback address, to be proxied.)

Figure 1 shows the dialog for setting up Firefox to use our example proxy.

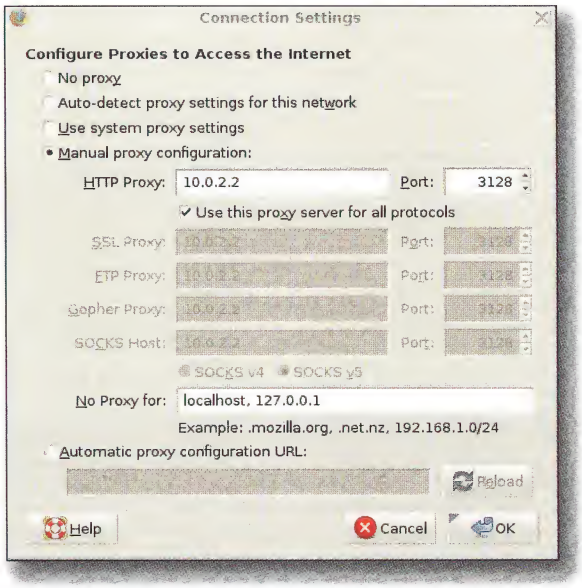


Figure 1. Setting Up Firefox to Use Proxies

In Figure 1, we've selected Manual proxy configuration and entered in an HTTP Proxy address (which can be either a hostname or IP address) of 10.0.2.2 and Port number 3128, which is Squid's default listening port for client connections. We've also selected the box to Use this proxy server for all protocols, resulting in the same values being copied automatically to the subsequent settings for other types of proxies.

We've left No Proxy for: at its default value of localhost, 127.0.0.1. The reason for not proxying connections to Web pages hosted locally on the client system is probably obvious, but you can additionally list URLs or IP addresses elsewhere on your local network that there's no need to use the proxy to reach.

At this point, you may be wondering, what does the connection between a client and a Web proxy look like? Is there some special protocol, or maybe a subset of HTTP commands or flags?

In fact, proxy connections are simpler than you may think. Normally, when you click on a hyperlink or enter a URL, your browser resolves the URL you typed or clicked on, using its own local DNS capabilities. It then takes the IP address and sends an HTTP/HTTPS request to that IP address, with the *original* (non-resolved) URL in the body of the request.

A proxied connection is the same without any DNS resolution. Your browser simply sends its

At this point, you may be wondering, what does the connection between a client and a Web proxy look like?

HTTP/HTTPS request to the proxy server without trying to resolve the URL. The body of that request is *identical* to the one it would otherwise send directly to the Web server you're trying to reach.

Instead of configuring your Web browser's proxy settings directly, if you use the GNOME desktop on your client test system, you can set global proxy settings that can, in turn, be used by Firefox and other Internet applications. Note, however, that the proxy settings you set in GNOME will be applied only to applications that are, in turn, configured to use system settings—for example, by selecting the option Use system proxy settings shown in Figure 1. Other applications will continue to use either their own proxy settings or no proxy at all.

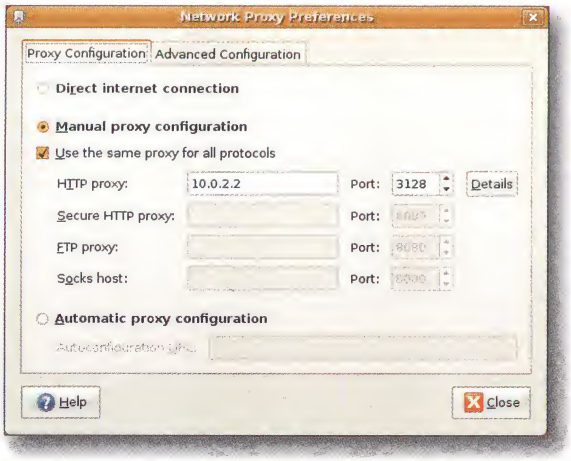


Figure 2. Setting Global Proxy Options in GNOME

GNOME's Network Proxy Preferences applet, which should appear in your System→Preferences menu, is shown in Figure 2.

It may seem like I'm spending a lot of ink explaining client-side configuration just for testing purposes, given that this is an article about building Squid servers. But, of course, the way you set up a proxy client for testing is the same as for one in production, so I would have had to explain this sooner or later anyhow.

In fact, future installments in this series may go further in covering client configuration topics. Autoproxy.pac files, for example (which is what

Squid's Performance Benefits

The Paranoid Penguin is a security column, so naturally, security is our primary focus in dealing with Squid (or it will be, once I've walked you through the basics of getting it up and running). But, you should be aware that Squid is not a security application per se. Squid's main purpose in life is to cache commonly accessed Web and FTP content locally, thereby both reducing Internet bandwidth usage and speeding up end users' download times.

The negative side of this is that Squid doesn't have as rich of a security feature set built in to it as commercial security-oriented Web proxies, such as BlueCoat and Sidewinder. In fact, Squid (years ago) used to ship with a default configuration that allowed completely open access.

The good side is that Squid can be configured, especially along with add-ons like SquidGuard, to provide some of the most important Web proxy security features. And, even if those features are your main reason for deploying Squid, you'll still enjoy the performance benefits of having commonly accessed Web content cached locally by Squid.

Seldom, in the security business, do we enhance end users' experience when we add security controls.

Figure 1's Automatic proxy configuration URL setting is for), can be very handy in managing very complex or very highly scaled proxy environments.

Once you've configured your test client system to use your Squid proxy, you can attempt to navigate to some Web page to see if everything works. It's a good idea to tail Squid's access log simultaneously. To do so, enter this command on your Squid system:

```
bash-$ sudo tail -f /var/log/squid/access.log
```

If browsing works but nothing zings by in this log-tailing session, your client-side configuration is incorrect—it isn't actually using the proxy. If browsing doesn't work, you may see some useful server-side message in the log-tailing session. Squid usually returns fairly useful messages directly to client browsers as well.

If things don't work, your browser session is simply timing out and nothing is showing up in access.log, try using the ping command from your client to your proxy and vice versa. If pinging doesn't work, the problem is at the network level and has nothing to do with Squid.

Conclusion

With any luck, at this point, chances are that everything works! Your Squid proxy software is installed, configured to accept only client connections from itself and from hosts on your local network, and it's hard at work proxying your users' connections and caching commonly accessed content. Not a bad day's work!

Not difficult, was it? Like most server applications, Squid's default configuration file is designed to maximize your chances for success, while minimizing the odds of your shiny-new Squid server being hacked. But, also like other server applications, there's certainly more that you *can* and *should* do to secure your Squid proxy than the default settings will do for you.

That will be our starting point next month. Among other things, we'll delve much deeper into Squid's Access Control List features to further harden Squid. Until then, be safe!■

Mick Bauer (darth.elmo@wiremonkeys.org) is Network Security Architect for one of the US's largest banks. He is the author of the O'Reilly book *Linux Server Security*, 2nd edition (formerly called *Building Secure Servers With Linux*), an occasional presenter at information security conferences and composer of the "Network Engineering Polka".

Resources

The Squid home page, where you can obtain the latest source code and binaries for Squid: **www.squid-cache.org**

The Ubuntu Server Guide's Squid Chapter: **https://help.ubuntu.com/8.10/serverguide/C/squid.html**

The Squid User's Guide: **www.deckle.co.za/squid-users-guide/Main_Page**

KYLE RANKIN

When Disaster Strikes: Attack of the rm Command

Can the `rm -rf /` command ever be tamed? Learn how to pick up the pieces when rm runs rampant on your filesystem.

The following is the continuation of a series of columns on Linux disasters and how to recover from them, inspired in part by a Halloween *Linux Journal* Live episode titled "Horror Stories". You can watch the original episode at **www.linuxjournal.com/video/linux-journal-live-horror-stories**.

Some commands on the command line are so blunt, so potentially devastating, that every time I use them, I pause for a moment before I press Enter. In my last column, I discussed one of my all-time favorites: dd (which could possibly stand for Destroy Data). Of course, as useful as dd is, I don't use it every single day, so even though I approach the command with reverence, you might argue it doesn't compare to the true master of data destruction: rm. Yes, dd can wipe out your hard drive in a few short keystrokes, but nothing really matches the compact destructive power of `rm -rf /`.

True, most people aren't bitten by that version of the command. Usually, it's its more sinister brother, `rm -rf ./` run from the wrong directory. The scene plays out something like this:

```
rm -rf ./
```

Clicking noises from the hard drive... "Hmm, that's taking longer than I tho...HEY!" Ctrl-C Ctrl-C Ctrl-C.

It's too late. By the time you noticed you ran that command in the wrong terminal, half of your home directory is gone. Now when I started out with Linux, I always was told in true UNIX form that when you rm a file, it is gone, and there is no way you can get it back. Undelete commands were for DOS users anyway—we Linux users knew better, right? Well, it turns out, we don't. Most Linux users I know have deleted the wrong files at least once in their lives. Now, the best protection against this is a backup (noticing a common thread in this series?), but if you don't have a backup, you aren't completely without hope. Everything you might have been told about the rm command isn't entirely true, and by the end of this article, you'll find that Linux does have an undelete of sorts.

Everything you might have been told about the rm command isn't entirely true, and by the end of this article, you'll find that Linux does have an undelete of sorts.

Free Space Isn't Free

To understand how to recover a deleted file, it's important to understand what rm does. When rm deletes a file, it essentially adds those blocks to the available free space on that filesystem. Unless you use a tool like shred, the data in those blocks stays intact until another file overwrites them. Blocks aren't reused in any date order, so some freed blocks might stay on the system for days, weeks or even years before they are reallocated to a new file, while others could be reused almost immediately.

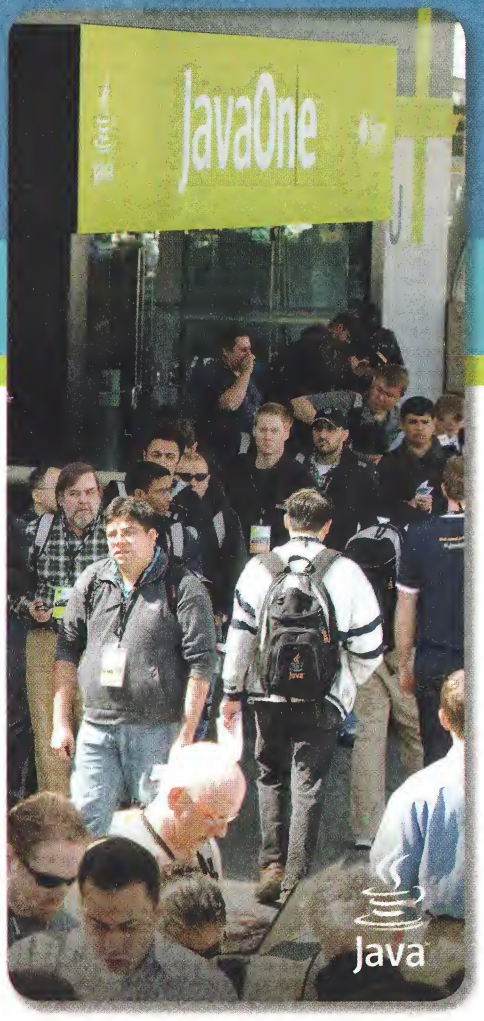
Because a Linux system writes files constantly, time is against you when you accidentally delete a file. The first thing you should do if you delete important files is unmount that filesystem. If you can't easily unmount the filesystem, shut down the system. Or, if the files are extra important, you might even pull the plug to ensure no other files are written to disk.

Forensics to the Rescue

It turns out that accident-prone Linux users aren't the only ones who want to recover deleted files. In fact, deleted file recovery is particularly useful for forensics, as attackers might try to delete files to cover their tracks. Forensics tools work with the filesystem on a low level as it is, because they try to gather data traditional tools might miss.

To recover deleted files, you need to install

WHERE
will YOU be on June 2?



JavaOne℠

JavaOne℠ Conference | June 2–5, 2009

The Pavilion: June 1–4, 2009, The Moscone Center, San Francisco, CA

The JavaOne℠ conference brings together developers, technology enthusiasts, and industry luminaries from around the world. It's your chance to learn, grow, and network with the vast—and growing—open technology community.

This year's JavaOne conference offers even more opportunity to grow your skills. You can:

- Connect with developers from more than 70 countries
- Hear from expert speakers on the hot topics you care about most
- Choose from a wide variety of targeted tracks, labs, and BOFs
- Get tips and best practices from technology creators and evangelists for areas such as next-generation Web services and cloud platforms
- Experience groundbreaking technologies—hands-on in our Pavilion

Nowadays money's tight. That's why it's more important than ever to attend the one conference that delivers everything you want to see, learn, and experience — all under one big roof. And that's the JavaOne conference.

Save Your Spot—Register Today!

By the way, check out CommunityOne, Sun's conference on open-source innovation and implementation, colocated with the JavaOne conference. **developers.sun.com/events/communityone**

Save $200
on Conference registration!

Register by April 22 at
java.sun.com/javaone

Cosponsored by

Sony Ericsson

Sun microsystems

sleuthkit. Most distributions these days offer it as a package; otherwise, you can download the source from the project's Web site. It may go without saying, but *don't install sleuthkit on the filesystem you are recovering!* If you need to recover files from the root filesystem, this may mean you have to take the hard drive to a second system or use a rescue disk like Knoppix that includes sleuthkit.

Once you have sleuthkit installed, you need to get a second disk that is large enough to store any files you want to recover. Unlike some other recovery methods, with sleuthkit, you don't have to create a complete image of the free space, so you won't need nearly as much storage. You can use the df tool to see how much free space you have:

```
$ df -h
Filesystem     Size  Used Avail Use% Mounted on
/dev/sda1      9.4G  7.0G  2.0G  79% /
/dev/sda3       20G   17G  3.6G  83% /home
```

In this case, I have around 2GB of space on my / partition and 3.6GB in /home to which to restore files. For this example, let's assume I have connected the recovery filesystem to this machine, and it has shown up as /dev/sda1. Be sure not to mount this filesystem. Or, if your machine automatically mounted it, be sure to unmount it before you continue, so you won't write to it accidentally. Because /home has more free space, I will recover to it, so I create a directory to store the recovered files and then use the sleuthkit fls (forensic ls) command to create a list of all the deleted files it can find on /dev/sda1:

Because a Linux system writes files constantly, time is against you when you accidentally delete a file.

```
$ mkdir ~/recovery
$ sudo fls -f ext -d -r -p /dev/sdb1 \
            > ~/recovery/deleted_files.txt
```

This command might take some time, depending on how much free space it has to pore through. In the meantime, we can discuss what these different arguments mean. The fls man page goes into more detail, but the -f argument specifies what filesystem fls is scanning (ext is used for ext2 and ext3). If you are unsure what value to use, type `fls -f list` to see a complete list of filesystems. By default, fls can list all the files on a particular filesystem, but when you specify -d, it lists only deleted ones. The -r option turns on recursion, so it traverses all directories it finds, and the -p option outputs the full path to each file. Without -p, if multiple files have the same name, it might be difficult to tell them apart. Finally, you list the partition you want fls to scan.

Once fls completes, you can open ~/recovery/deleted_files.txt to see a complete list of all the deleted files on the filesystem. It will look something like the following:

```
d/d * 944680:    home/kyle/.mutt
r/r * 943542:    home/kyle/.muttrc
r/r * 910452:    home/kyle/may_lj_article.txt
```

The first field tells you whether the file is a directory (d/d) or a regular file (r/r). Next is an inode number for the file, and then finally, you see the path to the file. Let's say, for this example, I want to recover the /home/kyle/may_lj_article.txt file. I then would use the sleuthkit icat tool to recover it. The icat program is a special version of cat that takes inodes as arguments. In this case, I would specify the inode 910452:

```
$ sudo icat -f ext -r -s /dev/sdb1 910452 \
              >~/recovery/may_lj_article.txt
```

As with fls, this might take some time to complete. You can read about all of its arguments in the icat man page, but here I use -f to specify the filesystem type like with fls. The -r option tells icat to go into a special recovery mode it uses for deleted files. The -s option causes icat to output the full contents of any sparse files it finds. Finally, I specify the partition to recover from and the inode to recover. Once the command completes, I can open ~/recovery/may_lj_article.txt and see whether it was able to restore it.

This method works fine when you need to recover only a few files, but what if you need to recover hundreds? Well, if you search on-line, you will find a number of different shell scripts people have written to recover all deleted files from fls output automatically. Below is one I originally found at **forums.gentoo.org/viewtopic-t-365703.html** and then improved a bit:

```
#!/bin/bash

DISK=/dev/sdb1 # disk to scan
RESTOREDIR=/home/kyle/recovery # directory to restore to

mkdir -p "$RESTOREDIR"
cat $1 |
```

```
while read line; do
    filetype=`echo "$line" | awk {'print $1'}`
    filenode=`echo "$line" | awk {'print $3'}`
    filenode=${filenode%:}
    filenode=${filenode%(*}
    filename=`echo "$line" | cut -f 2`

    echo "$filename"

    if [ $filetype == "d/d" ]; then
     mkdir -p "$RESTOREDIR/$filename"
    else
         mkdir -p "$RESTOREDIR/`dirname $filename`"
         icat -f ext -r -s "$DISK" "$filenode" \
              > "$RESTOREDIR/$filename"
    fi
done
```

Save this script under /usr/local/bin/restore. To use this script, replace the DISK and RESTOREDIR variables at the top of the script so they match your environment, give it executable permissions, and then run it with the fls output you created before as an argument. All of your recovered files will be wherever you set RESTOREDIR nested within their parent directories:

This method works fine when you need to recover only a few files, but what if you need to recover hundreds?

```
$ sudo chmod a+x /usr/local/bin/restore
$ sudo /usr/local/bin/restore ~/recovery/deleted_files.txt
```

Now, don't let this make you too comfortable with rm—there's no guarantee a particular file will be complete or even recovered at all. I still say the best policy is to have backups followed by a thoughtful pause before you press Enter on any recursive rm command.■

Kyle Rankin is a Senior Systems Administrator in the San Francisco Bay Area and the author of a number of books, including *Knoppix Hacks* and *Ubuntu Hacks* for O'Reilly Media. He is currently the president of the North Bay Linux Users' Group.

RotateRight's Zoom

The good folks at RotateRight informed us that their system-wide performance profiler for Linux, Zoom, has been updated to version 1.3. Zoom profiles are system-wide and precise down to the instruction level, and they capture backtraces. Zoom also analyzes and annotates code with specific tuning advice for most compilers and processors. The latest product update features a number of enhancements to help increase programmer productivity and optimize Linux application performance, both of which reduce costs by making software faster and more energy-efficient. These include support for Intel Atom and Core i7 processors, ability to show kernel source and assembly, support for external debug info files, calculation of symbol ranges when missing symbol information and several others. Zoom is available for Linux x86-64, i386 and PowerPC 64.

www.rotateright.com

Sans Digital's AccuSTOR AS212X2

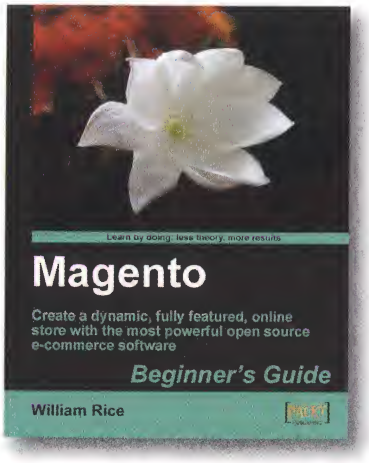
Sans Digital's new AccuSTOR AS212X2 Series is a 2U 12-bay SAS enclosure for mid-range and high-capacity storage environments. Sans Digital headlines the product as the first JBOD rackmount to overcome the lack of monitoring ability when utilizing a RAID controller card. A built-in selectable switch allows hardware monitoring via various popular brands of RAID controller interfaces. This new monitoring feature, says Sans Digital, "further simplifies the management process by allowing system administrators to access hard drive status, as well as power supply and cooling fan information". Data is protected by RAID protection provided by LSI, 3ware, Intel, Dell, ATTO, Areca or Adaptec SAS RAID controllers. The AS212X2 uses the latest SAS expander technology to connect up to 12 high-performance SAS drives or high-capacity SATA drives to the host computer using a single SAS cable, enabling a system bandwidth of up to 1,200MB/s.

www.sansdigital.com

Marvell Semiconductor's SheevaPlug

The SheevaPlug is one of the diminutive yet powerful devices in Marvell Semiconductor's "Plug Top Computing" initiative, a computing approach that features embedded, Linux-powered computers that plug in to electrical sockets. These devices, says Marvell, consume less than 5 Watts, can be left on all the time and "are capable of running network-based services that normally require a dedicated [PC]". These services include Web, e-mail and VPN servers hosted in homes and small offices. SheevaPlug features a 1.2GHz Marvell Sheeva CPU and 512MB each of Flash and DDR2 memory. Network connectivity is via Gigabit Ethernet; peripherals can be connected using USB 2.0. The SheevaPlug development kit contains the SheevaPlug and software tools needed to develop applications for the platform.

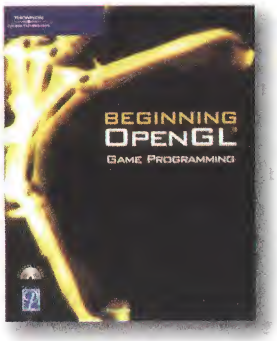
www.marvell.com



William Rice's *Magento Beginner's Guide* (Packt)

The open-source app Magento is one of the most evolved e-commerce solutions out there. For those starting a project from scratch, William Rice's new book, *Magento Beginner's Guide*, from Packt Publishing could be the ticket to success. Running on Apache-MySQL-PHP, Magento offers features such as multiple storefronts, templates and themes and multiple payment gateways (such as PayPal and credit cards). Because getting started with Magento can be daunting, Rice's book offers a step-by-step guide to getting a store up and running. It covers installation, configuration, populating a store with products, accepting payments, maintaining relationships with customers and fulfilling orders. After utilizing the book, readers will have a basic but complete and functional on-line store.

www.packtpub.com



Luke Benstead's *Beginning OpenGL Game Programming*, 2nd Ed. (Course Technology PTR)

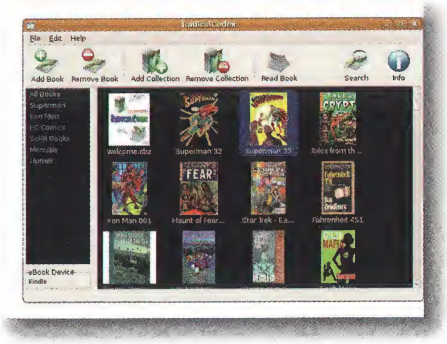
Realize your clandestine plan to develop the next runaway hit game with Luke Benstead's *Beginning OpenGL Game Programming*, 2nd Ed., from Course Technology PTR. The book provides "an easy-to-understand introduction to OpenGL, introducing all the basic elements of OpenGL as they apply to games", says the publisher. In addition, the new 2nd edition covers features found in OpenGL 3.0, the new and more efficient API that provides Direct3D 10 level graphics and is platform-independent. A companion CD-ROM features the source code used in the book, bonus chapters, games and the OpenGL Extension Library. Target readers are beginning game developers or programmers who are new to game development.

www.courseptr.com

Radical Breeze's RadicalCodex

Give your favorite superheroes a desktop home with Radical Breeze's RadicalCodex 1.0, an ebook and digital comic-book organizer and reader just for Linux. RadicalCodex enables users to read, bookmark, search and organize their entire e-comic library. The reader not only supports the most popular ebook and comic formats—such as PDF, TXT, CBR and CBZ—but it also exports ebooks to both the Amazon Kindle and the Sony PRS-505 via drag and drop. The CBR and CBZ formats are favored by many "indie" comic-book publishers. RadicalCodex is available for purchase from Radical Breeze's on-line store.

www.radicalbreeze.com



The Mono Project's Moonlight

Ancient are the days of a multimedia-handicapped Linux, thanks in part to applications like Moonlight, a newly 1.0 open-source project that gives Linux users access to Microsoft Silverlight content for the first time. It also plays Windows Media content. Moonlight is developed by the Mono Project, sponsored by Novell, and it works in tandem with the Banshee media player. Moonlight is part of a technical collaboration between Microsoft and Novell that offers a set of media codecs that bring optimized and licensed decoders for the Microsoft-based media formats. Developers also can write Rich Internet Applications for multiple platforms. Moonlight is available for all major Linux distros.

go-mono.com/moonlight

Appro's GreenBlade System

In an effort to save you money and save the planet at the same time, Appro has launched its GreenBlade System, which the company bills as an "open, green and affordable blade solution for mid-sized businesses". Based on Quad-Core AMD Opteron Processors, the GreenBlade is an energy-efficient solution that consolidates server, storage, network, power and simplified management capabilities. The solution comes in a 5U form factor and offers a variety of blade configurations with up to ten dual-processor server blades and 80 processing cores. Other features include up to 64GB of memory and 1.0TB of storage per compute blade, and up to four 1,625 Watt high-efficiency (90%+) power supplies per system. Appro's GreenBlade System also is part of the Appro Go-Green initiative that seeks to "address the HPC environmental challenges with performance-optimized and power-efficient solutions".

www.appro.com

Please send information about releases of Linux-related products to newproducts@linuxjournal.com or New Products c/o *Linux Journal*, PO Box 980985, Houston, TX 77098. Submissions are edited for length and content.

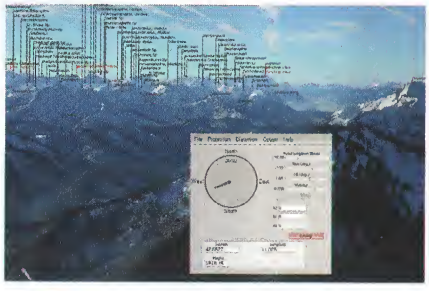
Fresh from the Labs

gipfel—Mountain Viewer/Locater

www.ecademix.com/JohannesHofmann/gipfel.html

This is definitely one of the most original and niche projects I've come across—and those two qualities are almost bound to get projects included in this section! gipfel has a unique application for mountain images and plotting. According to the Web site:

> gipfel helps to find the names of mountains or points of interest on a picture. It uses a database containing names and GPS data. With the given viewpoint (the point from which the picture was taken) and two known mountains on the picture, gipfel can compute all parameters needed to compute the positions of other mountains on the picture. gipfel can also generate (stitch) panorama images.



gipfel provides some amazing geological information when you position just two mountains.

Installation A source tarball is available on the Web site, and trawling around the Net, I found a package from the ancient wonderland of Debian. But, the package is just as old and beardy as its parent OS. Installing gipfel's source is a pretty basic process, so I went with the tarball. Once the contents are extracted and you have a terminal open in the new directory, it needs only the usual:

```
$ ./configure
$ make
```

And, as sudo or root:

```
# make install
```

However, like most niche projects, it does have a number of slightly obscure requirements that probably aren't installed on your system (the configure script will inform you). The Web site gives the following requirements:

- UNIX-like system (for example, Linux, *BSD)
- fltk-1.1
- gsl (GNU Scientific Library)
- libtiff

I found I needed to install fltk-1.1-dev and libgsl0-dev to get past `./configure` (you probably need the -dev package for libtiff installed too, but I already had that installed from a previous project). Once compilation has finished and the install script has done its thing, you can start the program with:

```
$ gipfel
```

Usage Once you're inside, the first thing you'll need to do is load a picture of mountains (and a word of warning, it only accepts .jpg files, so convert whatever you have if it isn't already a .jpg). Once the image is loaded, you either can choose a viewpoint from a predefined set of locations, such as Everest Base Camp and so on, or enter the coordinates manually. However, I couldn't wrap my head around the interface for manual entry, and as Johannes Hofmann says on his own page:

> ...gipfel also can be used to play around with the parameters manually. But be warned: it is pretty difficult to find the right parameters for a given picture manually. You can think of gipfel as a georeferencing software for arbitrary images (not only satellite images or maps).

As a result, Johannes recommends the Web site **www.alpin-koordinaten.de** as a great place for getting GPS locations, but bear in mind that the site is in German, *und mein Deutsch ist nicht so gut*, so you may need to run a Web translator.

If you're lucky enough to get a range of reference points appearing on your image, you can start to manipulate where they land on your picture according to perspective, as overwhelming chance dictates that the other mountain peaks won't line up immediately and, therefore, will require tweaking.

If you look at the controls, such as the compass bearing, focal length, tilt and so on, these will start to move the reference points around while still connecting them as a body of points. Provided you have the right coordinates for your point of view, the reference points should line up, along with information on all the other peaks with it (which is really what the project is for in the first place).

gipfel also has an image stitching mode, which allows you to generate panoramic images from multiple images that have been referenced with gipfel. As my attempts with gipfel didn't turn out so well, I include a shot of Johannes' stunning results achieved from Lempersberg to Zugspitze in the Bavarian Alps, as well as one of the epic panoramic shots as shown on the Web site. Although this project is still a bit unwieldy, it is still in development, and you have to hand it to gipfel, it is certainly original.

Also included in gipfel is the ability to stitch several images together for amazing panoramic shots like this.

Widelands—Real-Time Strategy

xoops.widelands.org

I covered this game only briefly in the Projects at a Glance section in last month's issue, so I'm taking a closer look at it this month. *Widelands* is a real-time strategy (RTS) game built on the SDL libraries and is inspired by *The Settlers* games from the early and mid-1990s. *The Settlers I* and *II* games were made in a time when the RTS genre was still in its relative infancy, so they had different gameplay ideals from their hyperspeed cousins, where a single map could take up to 50 hours of gameplay.

Widelands' **main emphasis is on base building and how you build it.**

Widelands **also has a lot of different settings and stories available to keep things interesting.**

Thankfully, *Widelands* has retained this ideal, where frantic "tank-rush" tactics do not apply. *Widelands* takes a much slower pace, with an emphasis not on combat, but on building your home base. And, although the interface is initially hard to penetrate, it does lend itself to more advanced elements of base building, with gameplay mechanics that seem to hinge on not necessarily what is constructed, but *how* it is constructed.

For instance, the ground is often angled. So, when you build roads, you have to take into account where they head in order for builders to be able to transport their goods quickly and easily. Elements such as flow are just about everything in this game—you almost could call it feng shui.

Installation If you head to the Web site's Downloads section, there's an i386 Linux binary available in a tarball that's around 100MB, which I'll be running with here. For masochists (or non-Intel machines), the game's source is available farther down the page.

Download the package and extract it to a new folder (which you'll need to make yourself). Open a terminal in the new folder, and enter the command:

```
$ ./widelands
```

If you're very lucky, it'll work right off the bat. Chances are, you'll get an error like this:

```
./widelands: error while loading
  shared libraries: libSDL_ttf-2.0.so.0:
cannot open shared object file: No such
  file or directory
```

I installed libSDL_ttf-2.0-dev, which fixed that, but then I got several other errors before I could get it to start. I had to install libSDL_gfx.so.4 and libsdl-gfx1.2-4 before it worked, but *Widelands* relies heavily on SDL (as do many other games), so you might as well install all of the SDL libraries while you're there.

Usage Once you're in the game, the first thing you should do is head to the Single Player mode, and choose Campaign to start, as there's a good tutorial, which you *will* need. While the levels are loading, hints are given to you for when you get in the game, speeding up the learning process.

Controls are with the mouse and keyboard. The mouse is used for choosing various actions on-screen, and the keyboard's arrow keys let you move the camera around the world. Left-clicking on an insignificant piece of map brings up a menu for all of the basic in-game options. Right-clicking on something usually gets rid of it.

From here on, the game is far too complex to explain in this amount of space, but it's well worth checking out the documentation and help screens for further information. Once you've finished the intro campaign, check out the game's large collection of single- and multiplayer maps. You get a choice of multiple races, including Barbarians, Empire and Atlanteans, coupled with the ability to play against the computer or against other humans (or a close approximation). It also comes with a background story to the game, and if you spend your Saturday nights playing *World of Warcraft* instead of going to the pub, I'm sure you'll find it very interesting.

Delve into this game, and there's much that lies beneath the surface. It has simple things that please, like how the in-game menus are very sophisticated and solid, with none of the bugginess you get in many amateur games. But, it's the complete reversal of hyperspeed in its gameplay that I really love. I always want to get back to building my base when playing most RTS games, but I'm constantly drawn away by fire fights. This game lets you keep building, and places serious emphasis on how you do it.

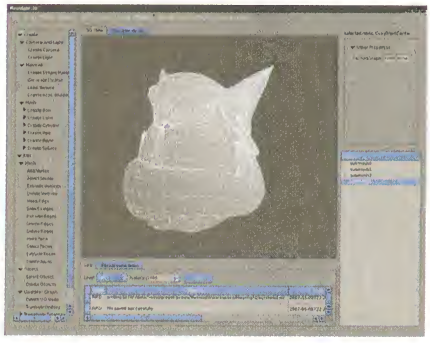
The Web site also has add-ons, such as maps, music and other tribes, along with an editor, artwork and more, so check it out. Ultimately, *Widelands* is a breath of fresh air in an extremely stale genre, whose roots ironically stem from way back in the past in RTS history. Whether you're chasing a fix of that original *Settlers* feel or just want a different direction in RTS, this game is well worth a look.

Moonlight|3D—3-D Image Modeling

www.moonlight3d.eu

This last project looks really cool and impressed me, but I'm afraid documentation is nonexistent, so hopefully some of you folks at home can help these guys out. According to the Freshmeat page:

> Moonlight|3D is a modeling and animation tool for three-dimensional art. It currently supports mesh-based modeling. It's a redesign of Moonlight Atelier, formed after Moonlight Atelier/Creator died in 1999/2000. Rendering is done through pluggable back ends. It currently supports Sunflow, with support for RenderMan and others in planning.



Some great results from someone who actually knows how to use Moonlight|3D.

The Web site sheds further light on the project, which states one of its goals as: "In order to speed up the progress of our development efforts, we open up the project to the general public, and we hope to attract the support of many developers and users, bringing the project forward faster."

Installation In terms of requirements, the only thing I needed to install to get Moonlight running was Java, so thankfully, the dependencies are fairly minimal. As for choices of packages at the Web site, there's a nightly build available as a binary or the latest source code (I ran with the binary). Grab the latest, extract it to a local folder, and open a terminal in the new folder. Then, enter the command:

```
$ ./moonlight.sh
```

Provided you have everything installed, it now should start. Once you're inside, I'm sorry, I really can't be of much help. There are the usual windows in a 3-D editor for height, width, depth and a 3-D view, and on the left are quick selection panes for objects, such as boxes, cones, spheres and so on (actually, the pane on the left has access to just about everything you need—it's pretty cool). Scouting about, a number of cool functions really jumped out at me, like multiple preview modes; changeable light, camera sources and positions; and most important, the ability to make your own animations. If only I could find a way to use them.

This project really does look pretty cool, and it seems to be a decent alternative to programs like Blender, but there honestly is no documentation. All links to documentation lead to a page saying the documentation doesn't exist yet and provides a link to the on-line forums. The forums also happen to have very little that's of use to someone without any prior knowledge of the interface, and I assume all those already on the forum are users of the original Moonlight Atelier. Nevertheless, the project does look interesting and seems to be quite stable. I look forward to seeing what happens with this project once some documentation is in place.■

John Knight is a 24-year-old, drumming- and climbing-obsessed maniac from the world's most isolated city—Perth, Western Australia. He can usually be found either buried in an Audacity screen or thrashing a kick-drum beyond recognition.

Brewing something fresh, innovative or mind-bending? Send e-mail to newprojects@linuxjournal.com.

EtherDrive®

The AFFORDABLE Network Storage

Now Supporting VMware® ESX 3.5

Fibre Channel speeds at Ethernet prices!

Is your budget shrinking while your network storage needs are growing? Are you suffering from "sticker shock" induced by expensive Fibre Channel and iSCSI storage area network solutions? EtherDrive® SAN solutions offer Fibre Channel speeds at Ethernet prices! Starting at just $1,995 for a 4TB system, EtherDrive® is the ***affordable*** storage area network solution. With sustained access speeds from 200MBytes/sec to over 600MBytes/sec, EtherDrive® SAN solutions are ***fast***. From a 4TB single storage appliance to multi-PetaByte system by simply adding more storage appliances, EtherDrive® SAN solutions are ***scalable***. From a single storage appliance to a network of sophisticated virtualized storage LUNs, EtherDrive® SAN solutions embrace ***virtualization***.

Coupling Ethernet technology with SATA hard disk drives, EtherDrive® SAN solutions exploit commodity components to deliver ***affordable, fast*** storage area network solutions that keep more green in your wallet! Whether you use your own SATA compliant disk drives or our certified enterprise class disk drives, you are in control! EtherDrive® SAN solutions accept standard SATA hard disk drives. Ethernet and SATA disk drives - two proven technologies in one ***affordable, fast*** storage area network solution - EtherDrive®.

EtherDrive® SAN solutions use the open ATA-over-Ethernet (AoE) lightweight network storage protocol. Simple. Easy to understand. Easy to use. AoE uses Ethernet to transport ATA disk commands without the burden of TCP/IP overhead, thereby enabling disk drives to become AoE devices connected directly to an Ethernet network. An AoE device can be a single physical disk or a logical device made up of multiple disks. An EtherDrive® SAN appliance is an AoE target device.

Finally, an ***affordable, fast*** storage area network solution for your VMware® ESX 3.5 installation. The EtherDrive® VMware ESX Host Bus Adapter empowers ESX with AoE technology to deliver EtherDrive® SAN solutions for your VMware ESX 3.5 installation.

Shipping EtherDrive® RAID solutions since 2004, Coraid boasts thousands of satisfied customers spanning a broad spectrum of the market including enterprise, government, educational institutions, and hosting service providers. ***Call today*** to order your EtherDrive® solution, and join the ranks of our thousands of satisfied customers!

Call 1.877.548.7200
or visit our website at
www.coraid.com
International: +1.706.548.7200

CORAID

vmware® | technology alliance **PARTNER**

ESX 3.5 compatible EtherDrive® HBA

LINUX-POWERED AMATEUR ROCKET GOES USB

The next Portland State Aerospace Society rocket, scheduled for first launch this summer, will have new hardware, including a switch from CAN to USB.

Sarah Sharp

In summer 2005, I stood on a sandy hill a couple miles east of Bend, Oregon. Through my binoculars, I could see people scattered in a distant ring around our 12-foot amateur rocket, waiting to take pictures when it launched. A mile away, I could see the tents and cars at ground control.

I was part of a recovery team for the Portland State Aerospace Society (PSAS). PSAS is a completely open-source aerospace engineering group. You can take our open-source software and open hardware designs from our Web site (see Resources) and make your own rocket. Our long-term goal is to guide our rocket into space actively and put a cube satellite into orbit.

Figure 1. Portland State Aerospace Society Rocket Launch (Photo Credit: Dave Sharp)

Figure 2. Rocket Launch, Part II (Photo Credit: Dave Sharp)

Figure 3. Rocket Launch, Part III (Photo Credit: Dave Sharp)

That summer day, we weren't going into orbit; we were just testing our latest rocket. Our rocket would launch, deploy its parachute at about 18,000 feet above the ground, and then drift safely to the ground, all the while spewing sensor data over our 802.11 wireless telemetry link. Once the rocket had landed, the recovery teams would use the GPS coordinates to find the rocket.

Over my 2-meter ham radio, I could hear Andrew Greenberg (PSAS's self-proclaimed "benevolent dictator") warning the bystanders at the launch site that the rocket motor was about to go live. The DTMF tones to arm the rocket followed.

"...3...2...1. We have liftoff!" The ground crew could see the streaming video from the rocket showing the ground become farther and farther away. The Java RocketView software displayed the rocket's sensor data: GPS coordinates, acceleration, rotation, pressure and the state of all the rocket's subsystems. Everything looked good.

I watched the rocket get smaller and smaller as it shot into the sky. The Linux flight computer on board the rocket would evaluate all the sensor data and decide when to deploy the parachute. The parachute needed to be deployed in the five-second window when the rocket reached its peak altitude (apogee), slowed down and started to fall downward.

At ground control, the crew watched the flight computer decide to deploy the drogue shoot. Everyone cheered, because the hard part of the flight was over. Or so we thought.

Five seconds later, the flight computer figured out that the rocket was still falling. It tried to deploy the main parachute, but it was still accelerating, as if the parachutes hadn't deployed. Something was wrong. Andrew frantically began to send the DTMF tones to the rocket for an emergency parachute deployment. The flight computer reported seeing the DTMF tones, but the rocket continued to plummet toward the ground.

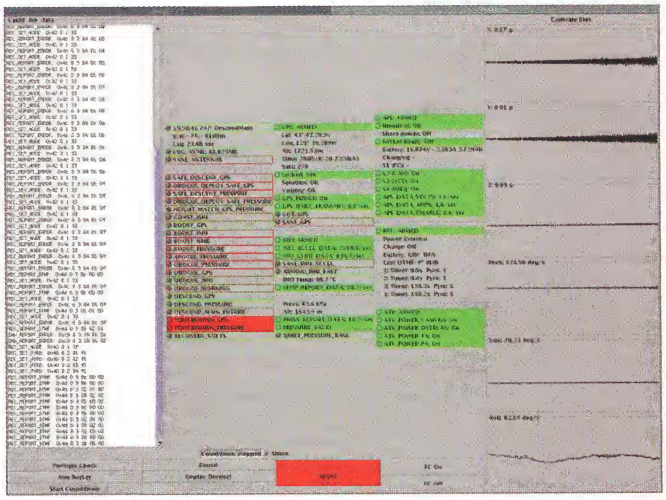


Figure 4. RocketView Screenshot (Photo Credit: Jamey Sharp)

Thirteen seconds later, the link to the flight computer was dead. The last known speed was more than 500mph, with a GPS reading about 1,000 feet off the ground. The depressed ground crew relayed the last-known latitude and longitude from RocketView.

Dave Allen, my fellow recovery team member, was eager to get to the rocket first. Dave and I got as close to the GPS coordinates as we could using the road and a four-wheel

OUR 12-FOOT ROCKET HAD BEEN COMPRESSED INTO A 3-FOOT PIECE OF TWISTED METAL.

Figure 5. Rocket Crash (Photo Credit: Sarah Sharp)

Figure 6. Maggie Emery Holding Baker the Sock Monkey, with Solomon Greenberg in the Background (Photo Credit: Sarah Sharp)

drive. Then we started hiking through the desert.

Finally, I spotted a glint of metal in the middle of a scrub brush. About a foot of rocket was sticking out of the ground. If we didn't have the GPS coordinates, it would have been impossible to find.

PSAS members showed up and we began to dig the rocket out. Our 12-foot rocket had been compressed into a 3-foot piece of twisted metal. The electronics were dust and bits of broken silicon. Amazingly, Baker, our sock monkey survived. He was a little squished, and his helmet was ripped, but he would fly another day.

Rising from the Dust: Redesigning the Rocket

After the 2005 crash, it would have been easy for PSAS to rebuild the rocket using this data. We toyed with the idea of rebuilding it exactly like the old rocket, but then "second system syndrome" set in. We just had to make the new rocket better than the old rocket.

The airframe team decided to redesign the airframe and the pyrotechnic parachute deployment system, as PSAS had

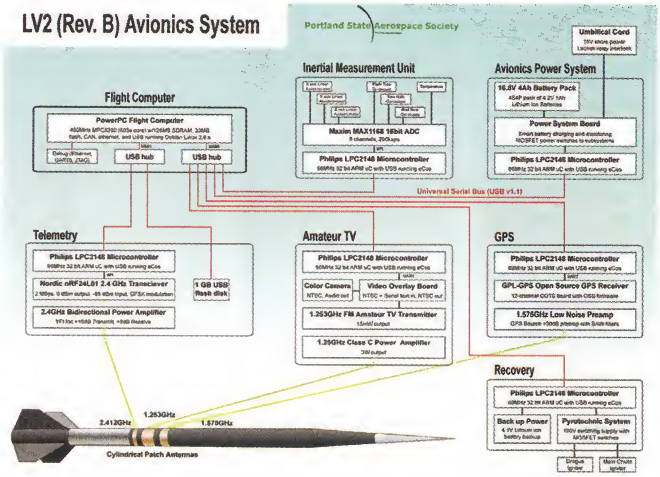


Figure 7. New PSAS Avionics (Credit: Andrew Greenberg)

concluded that was the point of failure for our launch. The avionics team decided to upgrade our flight computer from a 100MHz AMD Elan to a 400MHz Freescale MPC5200 (purchased with a grant from IBM).

The avionics team also wanted to upgrade the various avionics subsystems. We wanted the GPS, inertial measurement unit and all the other avionics sensor "nodes" to get data to the flight computer faster. The old rocket used 8-bit PIC microcontrollers that communicated over the Controller Area Network (CAN) bus. The avionics team wanted faster microcontrollers and a faster bus that was easier to develop software for.

Moving Toward USB

I was part of the Portland State University senior capstone project that was assigned the task of upgrading the avionics bus and sensor node microcontrollers. After much debate and argument within PSAS, we decided to replace the 1Mb CAN bus with a 12Mb full-speed USB. We chose a 32-bit ARM microcontroller, NXP's LPC2148 (see Resources).

The LPC2148 made the cut above the other 64-pin ARMs with USB because it already had an open-source library (LPCUSB) that would bootstrap the chip and control the USB peripheral. The main LPCUSB developer, Bertrik, was kind enough to let some PSAS members have commit access to the SVN repository, and PSAS has been contributing new features since then.

Choosing the LPC2148 also allowed us to pick from some very inexpensive hardware. An Olimex LPC2148 development board with USB, serial, JTAG and a built-in breakout area can be purchased for about $75. The Olimex JTAG programmers are about $50, and the free and open-source OpenOCD Project can be used to program the LPC2148 over the JTAG port. This makes it easy and cheap to build your own rocket avionics node at home.

You also can program LPC2148 to be whatever kind of USB device you want. The LPC2148 supports all four types of USB transfers and has enough Flash (32KB) and RAM (512KB) to support a moderate amount of code. Hardware hackers also will like the fact that it has I2C, SPI and plenty of GPIO pins. The LPCUSB library already supports several different USB

applications, such as a USB COM (serial) device and a mass storage device (Flash drive). These examples easily can be hacked to create custom USB devices.

Setting Up the LPC2148

If you want to start playing around with the LPC2148, you need to set up a development environment with a few different tools: an ARM-ELF cross compiler (for compiling code on a Linux box to ARM machine code), install tools for downloading the binary to the LPC2148, install host-side software to talk to the board and (optionally) the Eclipse IDE to set breakpoints on the LPC2148 and step through the code.

Fortunately, Dave Camarillo and Kay Wilson made a set of scripts to install and download all the necessary software and bundled them into a git repository with the PSAS LPC2148 source code:

```
$ git clone git://git.psas.pdx.edu/git/lpc-kit.git
```

The examples in this article assume you cloned the git repository from your $HOME/git/ directory.

Read the installation directions in the Doc/ directory. The psas_lpc_setup.pdf describes the hardware setup and what the scripts are trying to install. The scripts assume you're running on a Debian or Ubuntu Linux box, but they easily can be modified to run on an RPM-based distro.

Once you've run the shell scripts in the Tools/ directory, you can compile and download the simple serial example in the Dev/2148/poke/src/ directory to the LPC2148. The whole process is documented in the "Example Programming" section of psas_lpc_setup.pdf. The document walks you through setting up the cables, making the sample code by using the Makefile in Dev/2148/poke/src/ and using OpenOCD to program the LPC2148 board.

When you plug the reprogrammed LPC2148 in to an RS-232 port into your computer, a TTY device is created. If you're using a straight-through serial cable, /dev/ttyS0 is used. If you're using a USB-to-serial adapter, /dev/ttyUSB0 is created. Then, you can use minicom, or any other terminal emulator, to talk to the LPC2148 board. The default minicom settings (115200 baud rate, 8N1) are fine.

The reprogrammed LPC2148 echoes back whatever you type and outputs messages when you press the round black buttons on the board. This simple example should allow you to verify your build environment and ensure that you can talk to your board over the serial port.

THE LPC2148 MADE THE CUT ABOVE THE OTHER 64-PIN ARMS WITH USB BECAUSE IT ALREADY HAD AN OPEN-SOURCE LIBRARY (LPCUSB) THAT WOULD BOOTSTRAP THE CHIP AND CONTROL THE USB PERIPHERAL.

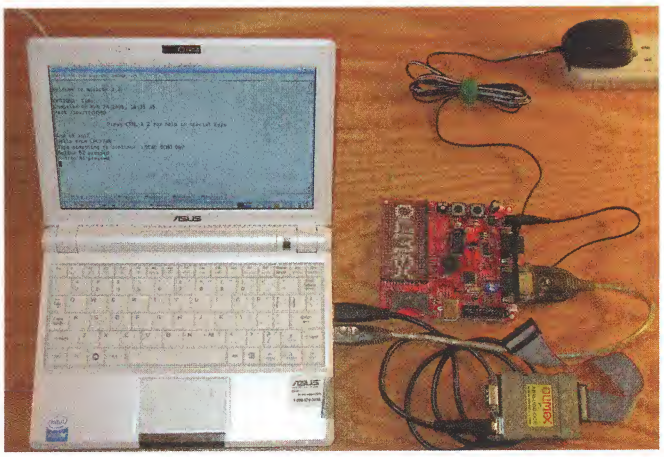


Figure 8. LPC2148 Example Setup (Photo Credit: Sarah Sharp)

LPC2148 USB Device

The more interesting project is to get the LPC2148 to communicate over USB. The LPC2148 supports four different USB transfer types: control, bulk, interrupt and isochronous. A USB device can have several data pipes, or "endpoints", that implement one of the transfer types. Each endpoint can either send data to the host (an IN endpoint) or send data from the host (an OUT endpoint). Control endpoints are bidirectional.

All USB devices must have one control endpoint over which to send their device descriptors. PSAS needed one other IN endpoint to send over periodically sampled sensor data, so we wanted either an interrupt or an isochronous IN endpoint. We always want to receive the latest data, so we chose the isochronous IN endpoint, because the host controller software will never attempt to retry a dropped isochronous transfer. Isochronous endpoints also could be used to turn the LPC2148 into a USB camera.

Dave and Kay recently added isochronous transfer and DMA support to the LPCUSB library. To try it out, you need to check out the latest code from the LPCUSB SVN repository:

```
$ svn co https://lpcusb.svn.sourceforge.net/svnroot/lpcusb lpcusb
```

I checked out version 177 into my $HOME/svn/ directory. Throughout these examples, I assume you use the same directories.

There should be an isochronous example in lpcusb/trunk/target/examples/ called isoc_io_dma_sample.c. This is a simple program for the LPC2148 that creates two isochronous endpoints. The IN isochronous endpoint sends a counter value into the host and then increments the counter. The OUT endpoint allows the host to control whether LED1 on the board is on or off.

To build the isoc example, change directories to lpcusb/trunk/target and type `make`. You now should have a file called isoc_io_dma_sample.hex in the examples directory.

Now you need to flash the .hex file to the LPC2148 board. You need to use the OpenOCD config file from the lpc-kit, and modify the OpenOCD script to download the correct .hex file.

First, copy the OpenOCD template script from lpc-kit:

```
$ cd ~/svn/lpcusb/trunk/target/examples/
```

```
$ cp ~/git/lpc-kit/Dev/2148/lpc-template/src/
➥oocd_flash_lpc2148.script .
```

Also, copy the OpenOCD config file into the LPCUSB examples directory:

```
$ cp ~/git/lpc-kit/Config/2148/openocd_lpc2148_v1257.cfg .
```

Now, modify the script to tell OpenOCD to send the isoc_io_dma_sample.hex file to the LPC2148. Change this line:

```
flash write_image template.hex 0x0 ihex
```

to:

```
flash write_image isoc_io_dma_sample.hex 0x0 ihex
```

Next, start the OpenOCD dæmon:

```
$ sudo ~/git/lpc-kit/LPC/2148/OCD/bin/openocd \
       -f openocd_lpc2148_v1257.cfg
```

From another terminal, Telnet into the OpenOCD port, and then tell OpenOCD to run the modified script:

```
$ cd ~/svn/lpcusb/trunk/target/examples/
$ telnet localhost 4444
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Open On-Chip Debugger
> script oocd_flash_lpc2148.script
```

If you've followed the instructions, LED2 on the Olimex board will start to blink incessantly, and you should see an OpenOCD message similar to the following:

```
wrote 9454 byte from file isoc_io_dma_sample.hex
                in 0.994377s (9.284629 kb/s)
```

Close the connection by pressing Ctrl-] and then Ctrl-D. Kill the OpenOCD dæmon in the other terminal by typing Ctrl-C. Remove the JTAG connector, press the LPC2148 reset button, and connect a USB cable from the Olimex board to your computer's USB port. Make sure to plug in to a root port, not through a USB hub. Some hubs have issues with isochronous transfers, so a direct connection is best. You can power the LPC2148 solely off USB bus power, but I left the 9V wall wart plugged in.

If you have CONFIG_USB_DEBUG turned on in your Linux kernel config, you will be able to watch the USB subsystem connect to the USB device as you plug it in:

```
$ sudo tail -f /var/log/kern.log
... usb 2-2: New USB device found, idVendor=ffff, idProduct=0005
... usb 2-2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
... usb 2-2: Product: USBSerial
... usb 2-2: Manufacturer: LPCUSB
... usb 2-2: SerialNumber: DEADC0DE
```

Type `sudo lsusb` to see which USB devices are connected to your system. You should see a device with an ID of ffff:0005. For me, it showed up as device 15:

```
$ sudo lsusb
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 015: ID ffff:0005
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

You can use the -v flag to examine the full device descriptors. This outputs all descriptors for all devices, so it's best to limit the output to the LPC2148 device with the -d <ID> option:

```
$ sudo lsusb -v -d ffff:0005
```

You should see two endpoint descriptors, one for an isochronous OUT endpoint and one for an isochronous IN endpoint.

Congratulations! The Linux kernel can initialize the LPC2148 USB device successfully. Unfortunately, there is no standard Linux USB kernel driver for this device. Instead, you need to compile and run a user-space program that uses the Linux kernel USB interface (usbfs) to talk to the device directly.

First, you need to have the libusb-dev package installed to get the usb.h header file for usbfs:

```
$ sudo aptitude install libusb-dev
```

Now, change directories into the lpcusb host-side code examples:

```
$ cd ~/svn/lpcusb/trunk/host/linux_isoc_sample/
```

Type `make`. This creates src/linux_usbfs_isoc_io_test, a binary that needs to run as root. Type `sudo src/linux_usbfs_isoc_io_test` to talk to the USB device. You will see lots of messages scroll by, similar to the following:

```
Bytes/second 1226
Input Length 4 number sent from device 0x3116D4 ret 0 status 0 flag 2
error_count 0 number_of_packets 1 actual_length 0 start_frame 614
usercontext -1077961592 iso_frame_desc[0].actual_length 0
iso_frame_desc[0].length 128 iso_frame_desc[0].status 0

Bytes/second 1228
Input Length 4 number sent from device 0x3116D5 ret 0 status 0 flag 2
error_count 0 number_of_packets 1 actual_length 0 start_frame 615
usercontext -1077961592 iso_frame_desc[0].actual_length 0
iso_frame_desc[0].length 128 iso_frame_desc[0].status 0
```

The start_frame is the USB bus "frame number" in which

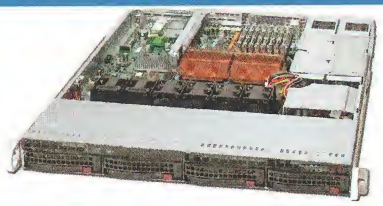
THE MORE INTERESTING PROJECT IS TO GET THE LPC2148 TO COMMUNICATE OVER USB.

The Straight Talk People℠
SINCE 1991

ABERDEEN

VMWARE CERTIFIED SOLUTIONS

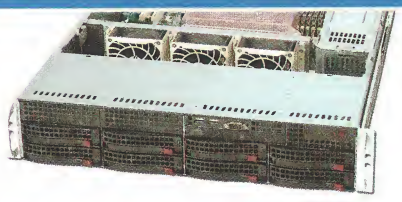
ABERDEEN STIRLING 123



1U Dual Xeon VMware Certified Server
- Up to two Dual-Core or Quad-Core Intel® Xeon® processors
- Up to 32GB 667/533MHz Fully Buffered ECC DDR2 SDRAM
- Up to 4 x Hot-Swap SATA or SAS Hard Drives
- Universal I/O allows for 3 expansion cards in 1U
- Pre-installed VMware® ESXi on Disk-on-Module
- 650W High-efficiency Redundant Power Supply
- 5-Year Warranty

Starting at **$1,999**

ABERDEEN STIRLING 229



2U Dual Xeon VMware Certified Server
- Up to two Dual-Core or Quad-Core Intel Xeon processors
- Up to 128GB 800/667/533MHz Fully Buffered ECC DDR2 SDRAM
- Up to 8 x 1TB (8.0TB) Hot-Swap SATA Hard Drives
- Up to 7 x Low-Profile Expansion Slots
- Pre-installed VMware ESXi on Disk-on-Module
- 700W High-efficiency Redundant Power Supply
- 5-Year Warranty

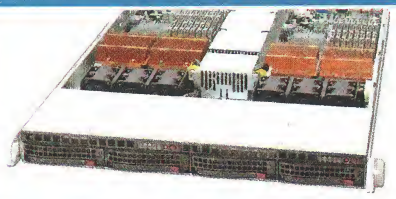
Starting at **$2,425**

ABERDEEN STIRLING 428T

4U Dual Xeon VMware Certified Server
- Up to two Dual-Core or Quad-Core Intel Xeon processors
- Up to 64GB 667/533MHz Fully Buffered ECC DDR2 SDRAM
- Up to 8 x 1TB (8.0TB) Hot-Swap SATA Hard Drives
- Up to 6 x Full Height Expansion Slots
- Pre-installed VMware ESXi on Disk-on-Module
- 800W High-efficiency Redundant Power Supply
- 5-Year Warranty

Starting at **$2,350**

ABERDEEN STIRLING 132T



1U Twin Node VMware Certified Server
- Up to two Dual-Core or Quad-Core Intel Xeon processors/node
- Twin Nodes allows for up to 4 processors & 16 cores in 1U
- Up to 64GB 800/667/533MHz Fully Buffered ECC DDR2/node
- Up to 2 x 1TB Hot-Swap SATA Hard Drives per node
- Pre-installed VMware ESXi on Disk-on-Module
- 980W High-efficiency Power Supply
- 5-Year Warranty

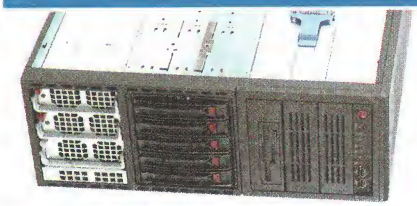
Starting at **$2,880**

ABERDEEN STIRLING 244

2U Quad Xeon MP VMware Certified Server
- Up to four Quad-Core or Six-Core Intel Xeon MP processors
- Quad Six-Core allows for 24 processor cores in 2U
- Up to 192GB 667/533MHz Fully Buffered ECC DDR2 SDRAM
- Up to 6 x Hot-Swap SATA or SAS Hard Drives
- Pre-installed VMware ESXi on Disk-on-Module
- 1200W High-efficiency Redundant Power Supply
- 5-Year Warranty

Starting at **$5,615**

ABERDEEN STIRLING 444



4U Quad Xeon MP VMware Certified Server
- Up to four Quad-Core or Six-Core Intel Xeon MP processors
- Quad Six-Core allows for 24 processor cores in 4U
- Up to 192GB 667/533MHz Fully Buffered ECC DDR2 SDRAM
- Up to 5 x Hot-Swap SATA or SAS Hard Drives
- Pre-installed VMware ESXi on Disk-on-Module
- 1200W High-efficiency Redundant Power Supply
- 5-Year Warranty

Starting at **$6,625**

ABERDEEN ISCSI XDAS

DAS VMware Certified Expandable Storage
- IP SAN Solution
- Single or Redundant Controller
- Expandable up to 64TB in a single array
- 2U/12 Bay and 3U/16 Bay Models available
- SAS or SATA Hard Drive Support
- Fault-tolerant Modular Hardware Design
- 5-Year Warranty

Starting at **$7,995**

ABERDEEN FIBRE XDAS

DAS VMware Certified Expandable Storage
- Hardware RAID5 and RAID6 engine by dedicated ASIC400
- Single or Redundant Controller
- Expandable up to 64TB in a single array
- 2U/12 Bay and 3U/16 Bay Models available
- SAS or SATA Hard Drive Support
- Fault-tolerant Modular Hardware Design
- 5-Year Warranty

Starting at **$8,495**

888-297-7409
www.aberdeeninc.com/lj029

Figure 9. PSAS 2009 Group Photo (Photo Credit: Sarah Sharp). Front row, left to right: Ken Zeigler, Jason Peterson, Andrew Greenberg, Daniel Heinlein, Nathan Bergey, Sarah Sharp. Middle row, left to right: Fletcher Hazlehurst and Frank Mathew. Back row, left to right: Ai Ling Chen, Jeremy Booth, Tim Brandon, Dave Camarillo, Kay Wilson, Mike Engstrom, Jamey Sharp, Josh Triplett, Theo Hill, Ian Osgood. Active PSAS members not pictured: Dan Kirkpatrick and Maria Webster.

the transfer started. A frame represents a one millisecond time period. As long as you see steadily incrementing start_frame numbers, you know the system isn't dropping isochronous packets. The hexidecimal "number sent from device" is the counter on the LPC2148 that is incremented when the interrupt handler is run and there's an isochronous IN transfer to send to the host.

The isochronous IN endpoint is working correctly if the start_frame and device counter are incremented at the same rate. They may be out of sync for the last couple transfers when you kill the program by pressing Ctrl-C. You also can tell whether the isochronous OUT endpoint is working if the LED1 on the board turns on and off every second.

Advanced LPC2148 USB Devices

This very simple code could be extended to make all sorts of USB devices. The isochronous IN endpoint could send sensor data like temperature, pressure or GPS readings. It also could send video, still frames or audio data. It even could be hooked up to a motion detector. The possibilities are endless with the Olimex's breakout board.

If you want to follow the Portland State Aerospace Society's development of LPC2148 USB avionics sensor nodes, join the psas-avionics list (see Resources).

PSAS hopes to do an airframe-only launch in Bend, Oregon, this summer. Our goal is to have working USB avionics nodes and a working Linux flight computer by October 2009. On October 2–4, the Arizona High Power Rocketry Association hosts the BALLS amateur rocketry event. If you're at the BALLS event in the Black Rock Desert of Nevada, or if you're still hanging around after Burning Man, stop by and say hello.■

Sarah Sharp graduated from Portland State University in 2007, but she continues to be an active member of the Portland State Aerospace Society. Sarah currently works at Intel's Open Source Technology Center as a Linux USB kernel hacker. Her blog can be found at sarah.thesharps.us.

Resources

Portland State Aerospace Society: **psas.pdx.edu**

NXP's LPC2148:
www.nxp.com/pip/LPC2141_42_44_46_48_4.html

LPCUSB Wiki: **wiki.sikken.nl/index.php?title=LPCUSB**

svcs.cs.pdx.edu Mailing Lists:
lists.psas.pdx.edu/mailman/listinfo

BALLS 18: **www.balls17.com**

2009 USENIX ANNUAL TECHNICAL CONFERENCE

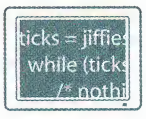
June 14–19, 2009
SAN DIEGO, CA

oin us in San Diego, June 14–19, 2009, for the 2009 USENIX Annual Technical Conference. USENIX nnual Tech has always been the place to present groundbreaking research and cutting-edge practices in wide variety of technologies and environments. USENIX '09 will be no exception.

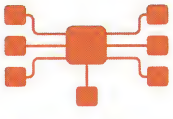
SENIX '09 topics include:


Security


Sysadmin



Coding


Networking


Open Source


Virtualization

oin the community of programmers, developers, and systems professionals in sharing solutions and resh ideas.

www.usenix.org/usenix09/lja

USENIX

THE CAMBRIDGE AUTONOMOUS UNDERWATER VEHICLE

A team from Cambridge designed and built an autonomous underwater vehicle for an annual Europe-wide competition. The AUVs will be tested by an underwater assault course that must be completed with no communication to or from the surface.

Andy Pritchard

Creating autonomous systems is a fascinating topic and has been ever since Isaac Asimov wrote about robotics in the early 1940s. Such a system can navigate unknown terrains, perform tasks and make decisions without assistance from humans. Lawn mowers and vacuum cleaners, able to operate without intervention, are simple examples of these concepts. Autonomous Underwater Vehicles (AUVs) are now becoming a major area of research and development with large companies investing in advancing this technology for both defense and academic purposes.

Several competitions have arisen from the recent interest, such as the Autonomous Underwater Vehicle Student Initiative (AUVSI) challenge, the Defense Advanced Research Projects Agency (DARPA) grand challenge and the Student Autonomous Underwater Vehicle Challenge-Europe (SAUC-E). All are aimed at encouraging student teams to develop solutions to some interesting problems.

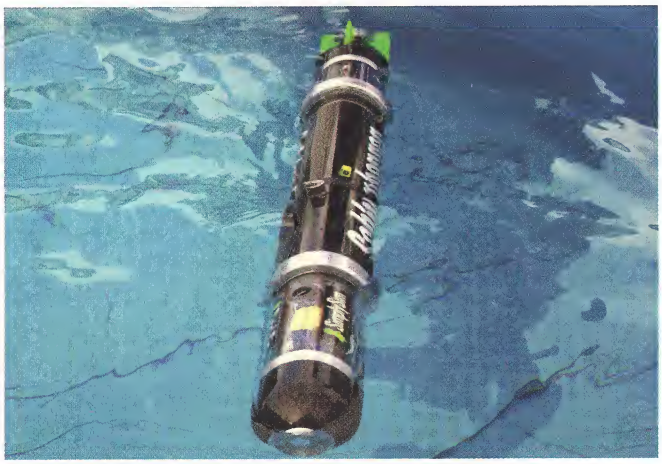


Figure 1. The Linux-Powered CAUV in the Water

The Cambridge Autonomous Underwater Vehicle (CAUV) team is a group of students from the University of Cambridge that has developed a Linux-powered AUV for the annual SAUC-E. The AUV must be able to complete an underwater assault course with no communication with the surface, external processors or outside intervention.

The Team

The CAUV team includes about 20 undergraduate students from around the university, studying computer science, engineering or natural sciences. Most students join to gain experience in the difficulties involved with team-oriented multidisciplinary design projects, with problems ranging from how to manage a team effectively to designing components that will operate correctly together in a system.

Figure 2. SAUC-E Competition

In previous years, we've done well in the competition. We took second place in 2007, and we won a prize for innovation in systems engineering from Direction des Constructions Navales Services (DCNS) in 2008. Preparations for the 2009 SAUC-E competition are underway, with high hopes of another strong result.

The AUV Hardware

Although the end-of-year competition is our short-term goal, we also have a long-term goal that heavily influences the design decisions we make. Part of our wider design aim for this project involves deployment into the Arctic through a bore hole in pack ice. To facilitate this, we chose a thin cylindrical design and avoided objects that would stick out of the hull, such as fins and external thrusters.

The chassis is small, lightweight and modular. The AUV is controlled by internal vector thrusters, two sets of two orthogonal thrusters that shoot jets of water to turn the vehicle and a propeller mounted at the stern. Most of the AUV is custom built in order to achieve the small size we require. Early on in the design process, we chose to split the AUV into five sections: the nose cone, the bow thrusters, the electronics racks, the stern thrusters and the tail cone.

The nose, tail and electronics sections are constructed from carbon fiber molded to a Myring Hull shape, allowing for minimum weight and maximum internal diameter. The nose cone contains a camera looking through a Perspex window, surrounded by a ring of bright LEDs to illuminate objects for the cameras in low light conditions. The bow thrusters' section houses a second camera and the vector bow thrusters along with associated electronics.

The electronics section houses the VIA EPIA PX10000G motherboard, which is supplemented with an array of dsPIC electronics to control the AUV and navigation equipment. The main sensory inputs are two orthogonal cameras and an inertial navigation system built by the CAUV team. The AUV's 12 2400mAhr Lithium polymer batteries make up the power core and offer a substantial working range.

We can estimate the best range and duration for the AUV using a basic fluid dynamics model of the AUV combined with data for power consumption and battery capacity. These calculations give a maximum range of 41km–51km at a cruise speed of 2.4m/s with a duration of 6–8 hours. The model also estimated the maximum speed at 4.2m/s.

Although our lightweight, high-density battery module allows for a good range, it does require careful management to avoid damage that can lead to explosions. This is where our custom-built battery management boards come in. Each battery has its own circuit that constantly monitors the temperature, charge and discharge rates. If any abnormal activity is detected, the battery is shut down and a log recorded in a central monitoring chip. At the 2008 competition, the battery management system was tested unintentionally when the AUV developed a leak, covering much of the electronics in water. Thankfully, the system worked perfectly, shutting down the batteries and protecting our electronics.

Waterproofing is a big concern for us, especially with the connectors that link the modules together. We fitted rubber O-rings to make the actual seal, but in order for them to be effective they need to be squashed against a smooth flat surface. To achieve this, we machined our connectors from aluminum with a 30-degree angle to act as the squash face for the O-ring. As some parts of the AUV need to be accessed more than others, we designed two types of connectors: quick and semi-permanent. The quick connector consists of a threaded aluminum sheath that screws over an aluminum ring attached to the receiving part of the hull, squashing the two parts of the

Figure 3. The CAUV out of the Water

connector together. The semi-permanent connectors use bolts that go up inside the AUV to compress the O-rings.

The AUV also is equipped with a mission-specific module (MSM) system so that extra hardware or sensors can be attached without major modification to the base of the AUV. We machined the MSM connector from an aluminum block, fitted bolts for the module to screw on to and provided a variety of wires connected up to the AUV electronics. Initially, these wires linked the mission-specific modules to the I2C bus of a dsPIC; however, we will change this to a more generic serial connection to the PICO ITX in the future. In previous years, this connection has been used to attach a marker dropping system required for the competition. This year, we plan to attach a much-needed sonar unit that has been kindly donated by Tritech International.

The Inertial Navigation System

The Inertial Measurement Unit (IMU) module is used to form a dead reckoning of the AUV position. It uses accelerometers, gyros and pressure sensors to calculate the position and orientation of the AUV. Due to the integration that is required to calculate the position, an error is built up over time, known as drift. To correct the drift, the camera will be used to calculate a more accurate position at a much lower refresh rate using a technique called simultaneous localization and mapping (SLAM). Ideally, we would use a GPS system to correct the error; unfortunately, these will not work when the AUV is submerged and also are restricted under the competition rules.

The IMU module has two main parts: an Inertial Navigation System (INS) and an autopilot. The INS circuit records data from the sensors and runs a continual integration loop in order to calculate the AUV's position. The autopilot circuit controls the propeller and four thrusters to move the AUV around the pool. If the vehicle is being operated in remotely operated vehicle (ROV) mode, the INS performs simple movement tasks, forwarding instructions from the main CPU. However, if in AUV mode with the autopilot active, it can use the current position (calculated by the INS circuit) to move to any destination set by the main CPU.

The navigation system, running on a dsPIC, communicates with the autonomy software, running on the PICO, using a simple serial protocol. To simplify the software, the board uses an FTDI chip to handle the USB-to-UART conversion. The protocol used sends simple command strings with checksum values attached to detect errors.

The AUV Software

In both 2007 and 2008, CAUV was the smallest robot at the competition, weighing in at less than 7kg. We use one of the world's smallest full-featured x86 motherboards to power the Ubuntu 8.10 operating system. Although the PICO board is small in size, it still is able to pack a punch with a 1GHz VIA C7 processor and 1GB of RAM, all of which is utilized by the onboard autonomy and image-processing software. Soon we would like to upgrade the processor to a Mobile ITX and possibly fit two boards in for some dual-processing fun.

The operating system used by the AUV is Ubuntu 8.10, chosen for its high reliability, low cost and ease of configuration. To save on some processing power and storage space, we have disabled or removed many of the default applications, such as GNOME. In previous years, we have used the Ubuntu Server edition and experimented with several different scheduling algorithms.

The primary storage device is a 4GB CompactFlash card, chosen for its low price, small size and energy efficiency compared to the equivalent mechanical device. All of these items are commodity goods that were bought off the shelf. Primarily used for cost and time reasons, there also is the added benefit that the community knowledge and support are outstanding.

Three modules make up the software: the decision-making software, the image-processing software and the navigation software. In 2008, we wrote all the software in Java, excluding the navigation software, as this is based on the dsPICs. For the 2009 software, we are porting our Java prototype to C++ and incorporating the OpenCV image-processing library to replace our custom image-processing system. The software modules are implemented separately and communicate via a network, allowing for one module to be run onshore during testing should we need to.

For the past two years, we have not had a sonar system to work with, so we have relied completely on vision. The AUV is fitted with two Webcams: one facing downward and one forward. The vision system must be able to recognize gates, buoys, tires and cones from any angle as well as differentiate the color of these objects in low light conditions in order to complete the assault course.

Our image-processing system is built on a series of filters joined together into a pipe, with filters including edge detection, Hough transforms and segmentation. This flexible system allows fast reconfiguration of processing methods should this be required.

Communication with the AUV is a vital part of the whole system. Without a reliable and usable way to relay information to and from the AUV, any data collected may lose its value. At the physical layer, we have two ways to communicate with the AUV. In the nose cone, we have an off-the-shelf Wi-Fi USB stick that can be used for remote surface communication. Naturally, this doesn't work so well underwater, so we have a second method for submerged communication. On the top of the AUV, we have a waterproof connector that is connected to the Ethernet port on the PICO. This means we can receive telemetry and image feeds from the AUV in real time, so long as we have a cable long enough.

Figure 4. And, the Winner is...the CAUV.

We have integrated a PlayStation II controller into the GUI that can be used when the AUV is tethered and in ROV mode. As well as being fun to play with, it creates a fast and effective remote control system.

The communication between the GUI and the AUV is a standard TCP/IP connection, with another of our own protocols running on top of this. The AUV is set up to send as much information as possible back to the GUI, where it is displayed graphically, if possible. The GUI contains a 3-D map of the path taken by the AUV and a series of graphs to plot telemetry. We hope to extend this map in the future to incorporate images taken by the AUV and show objects found by the AUV. To produce this map, we require a reliable, accurate stream of position data from our onboard navigation system, the inertial navigation system.

Conclusion

During the past two years, we have managed to build a solid base for the development of our AUV and gained a large amount of experience in the design process. Our long-term design goals are starting to be realized, and hopefully one day, we'll have a vehicle capable of withstanding Arctic conditions. Until then, we have an AUV that is a strong competitor in the SAUC-E competition that hopefully will match or better the results of previous years.

Acknowledgements

The Cambridge AUV team would like to say a massive thank you to our sponsors Schlumberger and Tritech International for their continued support. Without donations from companies such as these, many extra-curricular student-run projects, such as CAUV, wouldn't be possible.

If you'd like more information on Cambridge AUV or our sponsors, visit our Web site: **www.cambridgeauv.co.uk.**■

Andy Pritchard is the project manager of the Cambridge AUV Project, currently in his final year of the Computer Science Tripos at the University of Cambridge. He has been a member of the CAUV team for the past two and a half years.

Linux-Based 8mm Telecine

Got lots of 8mm film but no projector? Would you like to see those 30-year-old home movies your parents made when you were a kid one more time? Here's how a Linux system can be used to convert 8mm film to DVD movies.

FRANK PIRZ

Media Conversions, my business, converts videotape and slides to DVD. My customers often ask if I also can convert 8mm film. This is the story of my adventure into converting film to DVD. There are a number of ways to make a conversion. You can run the film through a projector and use a video camera to capture the images. Although, finding a working projector is difficult. Belts and rubber drive components dry up. Worse, 30-year-old rolls of film, some with splices, may no longer stand up to the stress of being projected at 18 frames/second (f/s). Plus, most video cameras run at 30f/s and will not synchronize with the projector.

Telecines have been used since the early days of broadcast TV to convert film to video. A number of Web sites describe DIY Telecine projects (see Resources). Generally, they either rebuild a projector and use a still camera, or they utilize a flatbed scanner and a custom film transport. Based on my research, I decided to build a Telecine using a flatbed scanner. The cost of entry is low, and scanners running at 3,000dpi or above are a commodity item. You can get started on the conversion software without the film transport, and you don't need custom optics. The downside, if you're not a programmer, is that you have to write all of your own software.

I decided early in the project that I wanted to use only open-source software tools. I hosted it on an Ubuntu Linux desktop system. I knew I would need a programming language with support for scanning, serial (or parallel) port communication, a math library and an image library. A plotting and drawing library also would be helpful during program development. I also wanted a language that offered ease of programming in higher-level constructs. I was familiar with C, but did not want to use it for this project, so instead, I decided to use Python. Python is easy to learn, it's well supported by the Linux community in both on-line forums and with numerous examples of code, and error handling and type checking/conversion are part of the language. Plus, the Python Imaging Library includes an interface to SANE for scanner support.

Film Transport

There are two parts to this project. One involves the software that processes the scanned film and makes a movie. The other part is the design of a film transport. The film transport is the harder part of the project, because it involves creating one-of-a-kind hardware. My transport design is based on reel-to-reel tape recorders popular in the 1960s (Figure a). It feeds film from a supply reel, across the scanner and winds it up on a take-up reel. A pair of spring-loaded idlers maintains film tension. A sprocket drive advances the film.

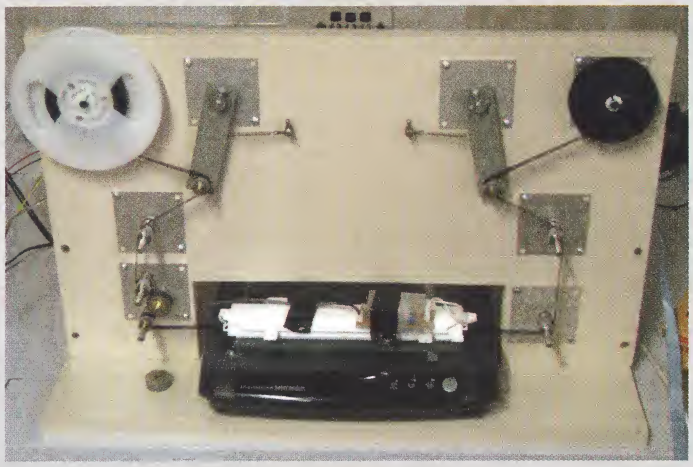


Figure a. 8mm Film Transport—Photo by Frank Pirz

The film transport is controlled by an embedded microprocessor. It takes commands from the Linux system over a serial port, and controls supply and take-up reel rotation and a sprocket motor for advancing the film. I was able to find both a program development and device programming environment as well as a C compiler for the Microchip PIC series of microprocessors all running under Linux. See Resources for the list of software tools used in this project.

I acquired an Epson Perfection 3490 photo scanner for the project. It has SANE drivers, a built-in backlight for film scanning and offers 3,200dpi resolution.

There are four steps to converting a roll of film: scan the film in segments, find the image frames in the segments, remove duplicate frames where the segments overlap and make a movie from the frames. I wrote three separate Python programs for the first three steps and used FFmpeg for the fourth. The software relies on cheap disk space. Frame files are copied from segment scans. Overlap removal makes a second, renumbered, copy of all of the frame files. This strategy allows each of those programs to be rerun with the same segment scans for debugging and program development.

The cost, for a 50-foot roll of film, is approximately 8GB of space for the segment scans and similar amounts of space for the log file (if debug is turned on) and each of the frame file sets. Files are written into subdirectories of the current directory and numbered sequentially. A root filename, given as a command-line argument, is used as a prefix. Scan data is written into the scans directory, and frame files are written to the frames directory. If logging is turned on, log files are written to the logs. If debug is left on (default setting), marked up copies of the scan files also are written to the logs. The markings show where the edges of the sprocket holes were found and the outline of the frame extracted. Finally, overlap-removed, renumbered frames are written to the movie directory.

The program for scanning film simply calls the SANE scanner interface, saves the scan data, advances the film and repeats for a count given as an argument on the command line. See the Film Transport sidebar for a description. You can do a project like this without a film transport, but it's tedious. Each scan takes about 80 seconds. Limits on the size of the backlight meant that I could use only about 7.7 inches out of the approximately 8.5 inches of scanner width. Allowing for overlap between the scans, a 50-foot roll of film will have about 90 scan segments and takes roughly two hours to scan.

To simplify the software, I made a film guide out of 10mm thick clear plastic film. I first aligned a steel ruler with the scanner axis, and I used GIMP to examine scans of the ruler edge. I moved it between scans until it was aligned to within approximately 50 pixels with the grid in GIMP. At 3,200dpi, 50 pixels is about 0.015 inches and more than adequate for this application. Then, I placed a piece of plastic against the ruler and glued it down with CyanoAcrylate glue. Once the glue was dry, I removed the ruler and used a piece of 8mm leader as a spacer to glue down a second guide. A sheet of glass placed over the guides keeps the film being scanned in alignment. With the film aligned with the scanner, no corrections for skewed images are necessary.

The program for finding frames actually is looking for sprocket holes. It's substituting software registration for mechanical registration of the film. Figure 1 shows a short piece of scanned film. The left-hand side is the original scan, and the right-hand side is the same scan converted to black and white (B&W).

Before we look for sprocket holes, we first find the top edge of the film. Given the alignment of the film in the guides, we could skip this step, but at this point, I'd rather not. The location of the top edge and knowing whether it's Regular8 or Super8 film (see the A Short History of 8mm Film sidebar), tells us approximately where the centerline of the sprocket holes will be.

The next step is to find the first sprocket hole. Because we are searching

A Short History of 8mm Film

The 8mm film format was developed by Eastman Kodak and released on the market in 1932 to create a home movie format that was less expensive than 16mm. The film spools actually contained 16mm film, which was exposed only along half its width. When the film reached its end, the camera was opened, and the spools in the camera were flipped. The same film was exposed along the side of the film left unexposed on the first loading. During processing, the film was split down the middle. This fit four times as many frames in the same amount of 16mm film. In 1965, Super8 film was released. It featured a bigger image area, resulting in a better quality image. It also moved the location of the sprocket hole and changed the hole size. Naturally, having two standards (see Resources) complicates both the software and hardware for an 8mm Telecine.

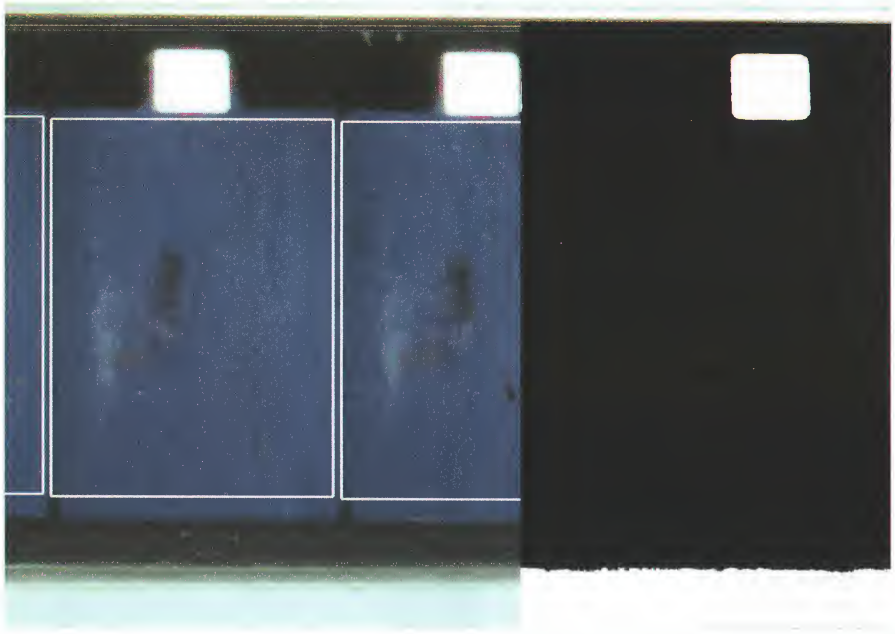


Figure 1. Sample 8mm Film—Used by Permission of Larry Stein

Figure 2. Illustrating Edge Failures—Used by Permission of Larry Stein

in a B&W image, we use a simplified correlation method. The search is done on a vertical line that spans the centerline we just found. If we find a white line, we add its value in to the correlation for that point. Black lines add zero. We have to look only at points inside the correlation window. Outside the window, the correlation value is zero. The process is sometimes called xor correlation, because addition replaces multiplication. The peak of the correlation function marks the edge of the sprocket hole.

With the edges of the first sprocket hole located, we know approximately where the centerline of the next sprocket hole should be. Simple line searches left and right from that centerline are used to find the next set of sprocket hole edges. The search ends at the last sprocket hole in the segment. Once we have found the left and right edges, we search up and down to locate the top and bottom edges. The film in Figure 1 shows the sprocket hole and frame markup after scanning.

If everything were that simple, we would be done. Naturally, it's not. The film segment in Figure 2 illustrates two problems. First, Kodak edge-marks its film. It says "safety film". Second, the image is not restricted to the frame area and has overlapped into the sprocket hole. Parts of the top and bottom edges of the sprocket hole have vanished in the B&W image. This will cause an edge-detection failure. There is a variety of heuristic methods to treat edge-detection failures. For left or right edge failures, I substitute the expected location based on the approximate sprocket center and the standard for the sprocket hole width. For top or bottom failures, my choice is to post-process the table of edges. When I find a missing edge or a run of missing edges, I average the edge values on either side of the gap and use the average as the location of the missing edge. It's important not to have abrupt changes in the sprocket hole locations, as this leads to visible jitter in the movie image.

Once all of the sprocket holes are found, the image frames are written to separate files in the frames subdirectory. The sprocket hole edge locations are

written out to the log file. Although I have not yet needed to do so, at some point, I expect to encounter a film segment where I cannot locate all the sprocket hole edges. Heuristic methods will take you only so far. It will be easier to use GIMP to find the elusive sprocket hole edges and edit the log file table with the correct coordinates. A modified version of the frame finding program could read in the corrected log file table and use that data to generate the image frames.

The images in the first movie I converted would get brighter and then get dimmer with a cycle of about 2–3 seconds. It was very visible and made the movie unusable. I'm scanning 45–46 image frames in each segment of film. At 18f/s, that's about 2.5 seconds of film. I'm using the film backlight removed from the cover of the scanner. It's a cold cathode fluorescent lamp with a white plastic diffuser in front of it. It was intended to backlight 35mm slides. It turns out that its light output is not uniform from end to end. Like most fluorescent lamps, it's slightly dimmer at the ends. Projector manufacturers go to significant lengths to make sure that the film is uniformly illuminated. See the link in Resources on Köhler Illuminators for more details. Replacing the lamp with a longer one didn't fix the problem.

An e-mail conversation with Richard J. Kinch led me to put illumination compensation into the software. I scanned a piece of neutral density film. Don't have any available? I didn't either. I cheated. I cut up a gray anti-static storage bag into strips. Two layers of the plastic film brought the resulting image into the middle of the gray scale. Then, I divided the scan into segments and sampled the image at the center of each segment. Not surprisingly, there was about a 30% variation from each end to the center. As the individual frame files are written out, a location-dependent compensation value is applied. This eliminated the illumination variation from the movie.

The final step is to remove the duplicate images where the scan segments overlap. The amount of overlap depends on how far you advance the film between scans. For this Telecine design, we have traded frame-accurate mechanical registration for software registration. We are not trying to be precise with the film advance. Typical scan segments overlap by two or more frames. The method for detecting a match between frames is called correlation. If two image files are identical, their correlation will be 1.0. If they differ, it will be less than 1.0. In practice, image frames of the same image scanned at either end of the scanner do not match precisely. The program for removing duplicates copies and renumbers frames to the end of the current segment. It matches the next-to-last frame of the current segment with the first five frames of the next segment. The frame with the highest correlation is the matching frame. The next segment becomes the current segment, and frame copying and renumbering begins with the frame after the best match. The process ends when there is no next segment.

At this point, we have converted the movie. It's just not in a format that is very usable. Some video editing software is capable of importing a sequence of image files and then writing out a movie file. Many do not. However, we are not really interested in editing the movie. We want to convert it and give it back to the customer. Using an editing program would be cumbersome. Instead, we use FFmpeg to read in the image frames and create a movie file in a format that's ready to burn on a DVD. A sample command line looks like this:

```
ffmpeg -r 18 -i movie/sam.%4d.tiff \
            -target ntsc-dvd -aspect 4:3 sam.mpg
```

Briefly:

- `-r 18` tells FFmpeg that the input file is at 18 frames/second.
- `-i movie/sam.%4d.tiff` implies the input files are named sam.0001.tiff, sam.002.tiff and so on.
- `-target ntsc-dvd -aspect 4:3` uses FFmpeg presets to create an .mpg movie file suitable for burning to DVD.
- `sam.mpg` is the generated movie file.

Consult the on-line documentation and the reference cited in the Resources section for more information. At this point, our job is done. A variety of Linux tools is available for authoring DVDs and burning DVD disks. Both are beyond the scope of this article.

This project demonstrates that customized, relatively sophisticated, image processing can be handled easily with Linux-based tools. It also describes embedded hardware development in a Linux environment. This project is continuing to evolve. Sprocket hole edges can be checked for abrupt changes. Once the frame files are extracted, there are opportunities for additional improvements. I have experimented with the ImageMagick toolset to sharpen the images and remove dust specks. The Python programs for image processing as well as the C code and other engineering documents for the film transport are both available from the *LJ* FTP site.■

Frank Pirz currently runs Media Conversions. He converts videotape, slides and now 8mm film to DVD format. His current interests include home theater, multimedia PCs and building robots. When he's not working, he's usually reading the latest *Star Trek* or *Star Wars* books. He can be reached at fpirz@media-conversions.net.

Resources

Code and Other Engineering Documents That Accompany This Article: **ftp.linuxjournal.com/pub/lj/listings/issue181/10373.tgz**

Flatbed Scanner Digital Telecine (FSDT) Project, by Richard J. Kinch: **www.truetex.com/telecine.htm**

Legacy Film to DVD Project, by Jim Carroll: **www.jiminger.com/s8/index.html**

Transferring Film to Video (Telecine), by Martin W. Baumgarten: **lavender.fortunecity.com/lavender/569/filmtovideo.html**

8mm2avi (a program to convert 8mm films to AVI) SmartSoftware Italia: **8mm2avi.netfirms.com/index.html**

A Homemade Telecine Machine, by Jan Demmendal: **www.movie2video.com**

MovieStuff (Roger Evans), sells equipment for film to video transfer (see also for good instructions about cleaning film): **www.moviestuff.tv**

Hub Adapters (Moment Catcher) Convert Super8 for Regular8: **www.momentcatcherproductions.com/page6.html#adapters**

Regular8 and Super8 8mm Film Specifications: **8mm2avi.netfirms.com/Specs.htm**

Köhler Illumination, by Michael Pate, Optical Short Course International: **www.loreti.it/Download/PDF/DMD/IlluminationSystemTypes.pdf**

SANE—Scanner Access Now Easy: **www.sane-project.org**

PythonWare Library—Includes PIL, Python Reference and Tutorial Documents: **www.pythonware.com/library/index.htm**

Python Imaging Library (PIL): **www.pythonware.com/products/pil**

NumPy (the fundamental package needed for scientific computing with Python): **numpy.scipy.org**

FFmpeg—Project Description: **ffmpeg.mplayerhq.hu/index.html**

Using ffmpeg to manipulate audio and video files, by Howard Pritchett (see the section on Basic Video Transcoding): **howto-pages.org/ffmpeg/#basicvideo**

ImageMagick: **www.imagemagick.org/script/index.php**

Film Sprockets—LaVezzi: **www.lavezzi.com/QA/LavSprocket.html**

Microchip (I used the PIC 16F876 chip for this project): **www.microchip.com**

PiKdev (a simple graphic IDE for the development of PIC-based applications): **pikdev.free.fr**

HI-TECH C PRO for the PIC10/12/16 MCU Family (Lite mode)—freeware: **www.htsoft.com/microchip/products/compilers/piccpro-modes.php**

PICList (a collection of people interested in the Microchip PIC): **www.piclist.com/tecHREF/microchip/index.htm**

PIC Sample Code in C: **www.microchipc.com**

Fun with the iRobot Create

Let your computer reach into the physical world with the iRobot Create.

ZACH BANKS

Very little in the Linux universe interacts directly with the physical world. Although you may have peripherals that allow you to work with the computer, the computer has no way to interact with you. This is easily solvable by creating a robot for it to control. iRobot, famous for its Roombas, has created an educational robot called the iRobot Create, based on the Roomba, that is incredibly easy to work with. The Create provides a simple base to extend upon with very little effort. Some people even have mounted an old laptop to the robot to allow mobility, but that is overkill for most situations. It's not hard to create a link between a Linux box and the Create, even though it lacks official support.

The easiest way to interact with the Create is through a serial link using the cable that comes with the robot. For some computers, you may need a USB-to-serial adapter; however, they are readily available for less than $15.

The connection will be a TTY serial, such as /dev/ttyS0, or if you are using a USB adapter, the connection most likely will show up as /dev/ttyUSB0.

In order to pass commands back and forth though the serial cable, the easiest tool to use is a serial port terminal. There are several versions of this type of software available. Here, I use gtkterm, a GUI terminal, but if you prefer CLI tools, both screen and minicom will work. After installing and launching gtkterm, you have to set the correct port under Configuration→Port. The port will be the device specified earlier, and if you are unsure which number to choose, you may have to try them all. The speed should be set to 57600 (baud). The other default settings (No parity, 8-bit, 1 stopbit and no flow control) are fine. I also prefer to turn on Local echo, which also is under Configuration and lets you see what you type.

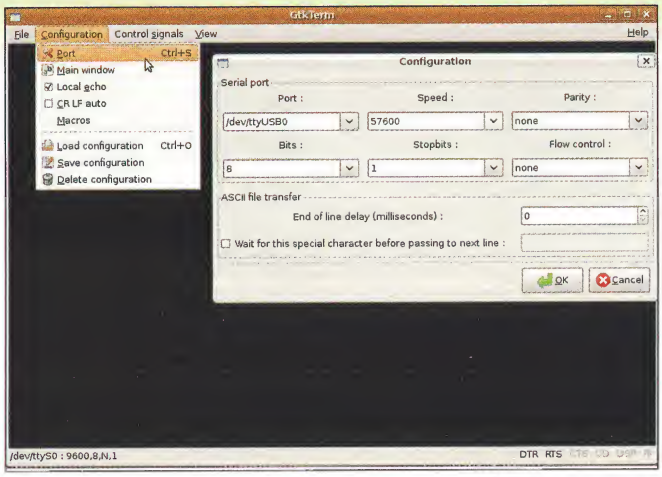


Figure 1. Configuration Options for gtkterm

To test the configuration, plug in the Create to charge and connect it to your computer. The terminal should start displaying lines such as the following every second:

```
bat:   min 0  sec 11  mV 16699  mA 566  deg-C 21
```

Unless you plan on mounting a computer to the robot itself, the serial cable will prove cumbersome as soon as the robot begins to move. To get around this, the robot needs to go wireless. Although 802.11 Wi-Fi has become ubiquitous on laptops, it is not common on embedded systems like the Create. Another candidate is Bluetooth, which also is becoming widespread; however, Bluetooth modules generally are expensive, have hit-or-miss Linux support and are very short-ranged. Recently, Maxstream's line of XBee radios have been gaining popularity in projects like this. They are very similar to Bluetooth modems and are better suited for this type of project.

All of the parts for this project can be purchased at SparkFun and are listed in Table 1. In addition to these items, you also will need some basic tools and supplies, such as a breadboard, wire and a soldering iron.

First, you need to configure your two XBee modules. To start, plug one of them in to the USB XBee explorer and connect it to your computer via USB cable (the USB XBee explorer is simply a serial-to-USB converter board that accepts an XBee module). Using gtkterm again, set it up to listen on a USB port (most likely /dev/ttyUSB0), and set the speed to 9600 baud. Type into the terminal `+++`, and the module should reply `OK`.

The module now is ready to be configured. Type in `ATID3330,DH0,DL1,MY0,BD6,WR,CN`, and after each comma, the module will reply with `OK`. Remove this XBee, and insert the other one. Again, type `+++`, and wait for the `OK` to enter into configuration mode. This time, however, configure it with `ATID3330,DH0,DL0,MY1,BD6,WR,CN`. Each module is configured to be on network 0x3330 and to send data directly to the other at 57600 baud. One module is connected to the computer, and the other to the Create. The modules are interchangeable—either one can be connected to the computer or the Create.

Next, build the circuit to connect the XBee with the Create serially. This circuit connects the 3.3-volt XBee to the 5-volt Create. To start, solder the two sockets into the XBee breakout board. The easiest way to do this is to place the sockets on the XBee module itself, flip it over, and place the breakout board on top.

After the sockets are soldered, remove the module and solder four wires to VCC, DOUT, DIN and GND. After that, solder four more wires to the male DB-25 connector on pins 1, 2, 8 and 21. The pins should be labeled, although the markings are faint. Next, break off two six-pin lengths from the strip of male header pins, and solder them to each side of the level converter. Again, it is easiest if you use the breadboard as a jig to hold the pins straight as you solder them. Finally, assemble everything according to the schematic (Figure 2) and/or the breadboard wiring diagram (Figure 3). The completed breadboard is shown in Figures 4 and 5.

Table 1. SparkFun BOM

Part #	Description	Quantity	Price Each
WRL-08664	XBee Module	2	$24.95
WRL-08687	USB XBee Explorer	1	$24.95
BOB-08276	XBee Breakout Board	1	$2.95
PRT-08272	XBee Socket	2	$1.00
BOB-08745	Level Converter	1	$1.95
PRT-00116	Male Header Pins	1	$2.50
COM-00526	3.3V Regulator	1	$1.95
COM-08375	0.1 uF Filtering Capacitor	1	$0.25
PRT-08287	Male DB-25 Connector	1	$0.95

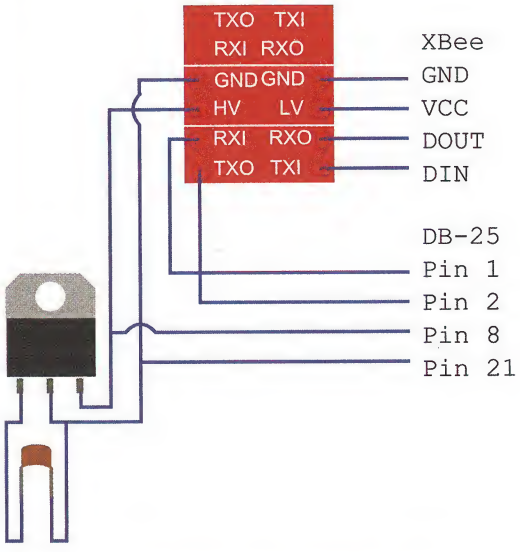


Figure 2. Schematic for the XBee/Create Interface

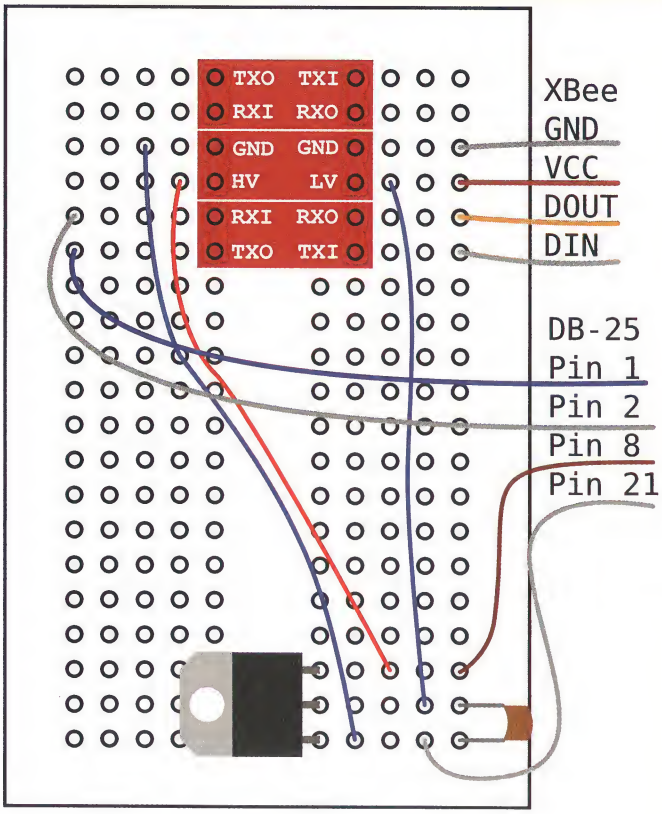


Figure 3. Breadboard Wiring Diagram for the XBee/Create Interface

Plug the DB-25 connector in to the Create's expansion port, and remove the command module if present. With the other XBee plugged in to your computer, set up gtkterm to communicate with it at 57600 baud. As before, plug the Create in to charge, and with luck, you will see some output on the terminal, and the RX light on the USB explorer should blink. If not, check your connections and configuration.

Even if you did not decide to go wireless, you still can control the Create in exactly the same way. The Create, and most Roombas, implement the iRobot Open Interface protocol, or OI for short. On the computer side, let's use Python to communicate with the Create using iRobot's implementation of OI in Python. This allows you to work on a higher level and not worry about opcodes and such. You will need pySerial and openinterface.py (see Resources). There is a small bug in openinterface.py that can make it difficult to work with on Linux. The simplest way to solve this is to run this sed command in the same directory as the file:

```
$ sed -ie "803s/ - 1//" openinterface.py
```

Alternatively, you can remove - 1 manually from line 803.

The library is easy to use—for example, to drive the Create forward at full speed, do this:

```
import openinterface as oi
PORT = "/dev/ttyUSB0"            # change to your serial port
bot  = oi.CreateBot(com_port=PORT,mode="full")
bot.drive_straight(500)          # drive forward, full speed
```

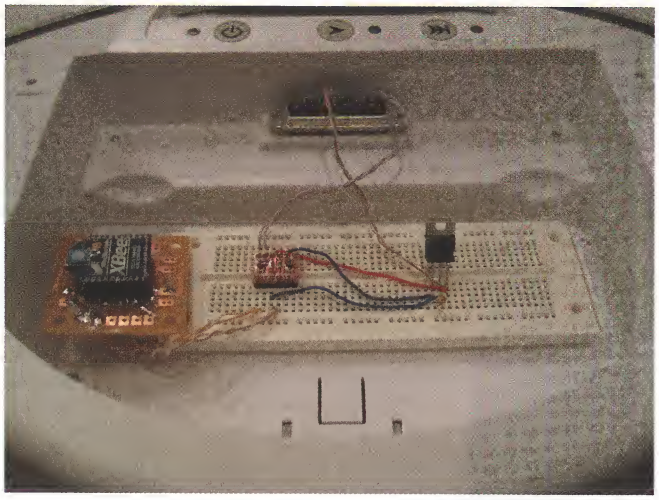


Figure 4. Overview of Electronics Components

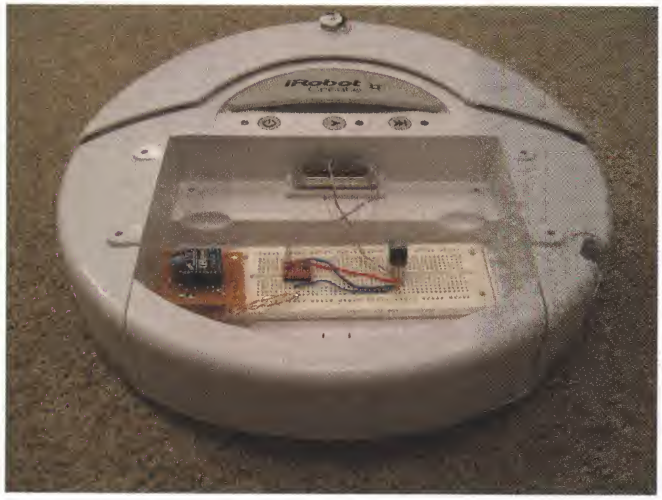


Figure 5. Overview of the Create with All Components Installed

In order to access sensor data, you need to request it. If you use `bot.stream_sensors()`, the Create will update the specified sensors in each argument automatically every 15 milliseconds. To stop, execute `bot.stop_streaming_sensors()`. Although you can specify manually which sensors you want to stream, it generally is easiest just to stream all of them.

Driving also is pretty simple. `bot.drive()` takes two arguments: speed and turning radius. Speed is an integer between 500 and –500, specifying the average speed of the wheels in millimeters per second, with negative values corresponding to going backward. Turning radius is a number between 200 and –200, specifying the radius of a turn in millimeters. Positive values turn left, and negative values turn right. There also are special methods that can be used for going straight and turning in place.

The following code uses sensor data to drive and maneuver around obstacles:

```
bot.stream_sensors(6)                    # packet 6 -- all sensors

while True:                              # loop forever
    if bot.sensors["bump-left?"]:        # is it pressed?
        bot.drive(-500, 10)              # spin to maneuverer
        bot.wait(5)                      # spin for 5 cycles
    elif bot.sensors["bump-right?"]:     # other direction
        bot.drive(500, 10)
        bot.wait(5)
    else:
        bot.drive_straight(500)          # otherwise, go forward
        bot.wait()                       # prevents excess cycling
```

You can access the Create's song-playing abilities very easily too, and you can store songs in the 17 available song slots. Use `bot.define_song()` to store a song. The first argument is the song slot where the song will be stored, and you also use this value later to play the song back. The rest of the arguments are notes, represented by tuples of pitch and length. Length is measured in 64ths of a second. Call `bot.play_song()` to play the song. I'm no musical genius, so hopefully you can write a better tune:

```
bot.define_song(1,              # index of song
                ("G1", 16),     # note tuples
                ("G2", 16),     # note, duration
                ("G3", 64),     # 64 = 1 second
                ("G9", 16))     # up to 100 notes
# ...snip...
bot.play_song(1)
```

To control the Create wirelessly with a joystick and Python, we can use pygame (the full details of the pygame joystick API are beyond the scope of the article; check the documentation for more information):

```
import pygame
from pygame import locals
pygame.init()
js = pygame.joystick.Joystick(0)      # create joystick
js.init()

import openinterface as oi
PORT = "/dev/ttyUSB0"                 # change to your serial port
bot  = oi.CreateBot(com_port=PORT,mode="full")

while True:
    if js.getAxis(0) > 0:
        turn = 1 - js.getAxis(0)
    else:
        turn = -(1 + js.getAxis(0))
    bot.drive(js.getAxis(1)*500, turn*200)
    bot.wait()
```

This code allows you to use a joystick (autodetected) to have primitive control over the Create. The x-axis value has to be manipulated so that when in a neutral position, the robot moves straight and does not spin.

Where to go from here? That's up to you. On the hardware side, you can attach additional hardware to the Create and control it through its digital inputs and outputs (see OI specifications for pin-outs). However, with just the base and some software, there still are tons of possibilities. For example, you could turn the Create into an alarm clock reminiscent of Clocky, the clock that drives around the room forcing you to get out of bed to shut it off. Or, if you are more mathematically inclined, you could use the the "distance" and "angle" sensors to map out a room.

Zach Banks is an experimenter who is stuck between hardware and software. He's glad to accept comments and questions at zjbanks@gmail.com.

Resources

SparkFun Electronics: **www.sparkfun.com**

iRobot Open Interface Specification: **www.irobot.com/filelibrary/pdfs/hrd/create/Create%20Open%20Interface_v2.pdf**

pySerial: **pyserial.wiki.sourceforge.net/pySerial**

openinterface.py: **createforums.irobot.com/irobotcreate/attachments/irobotcreate/Create_Support/792/2/openinterface.py**

pygame: **www.pygame.org**

HARDWARE

Control Your Home with Vera from Mi Casa Verde

Use Vera, a Z-Wave-based product, to automate your home, and let the computer turn the lights on and off for you.

DANIEL BARTHOLOMEW

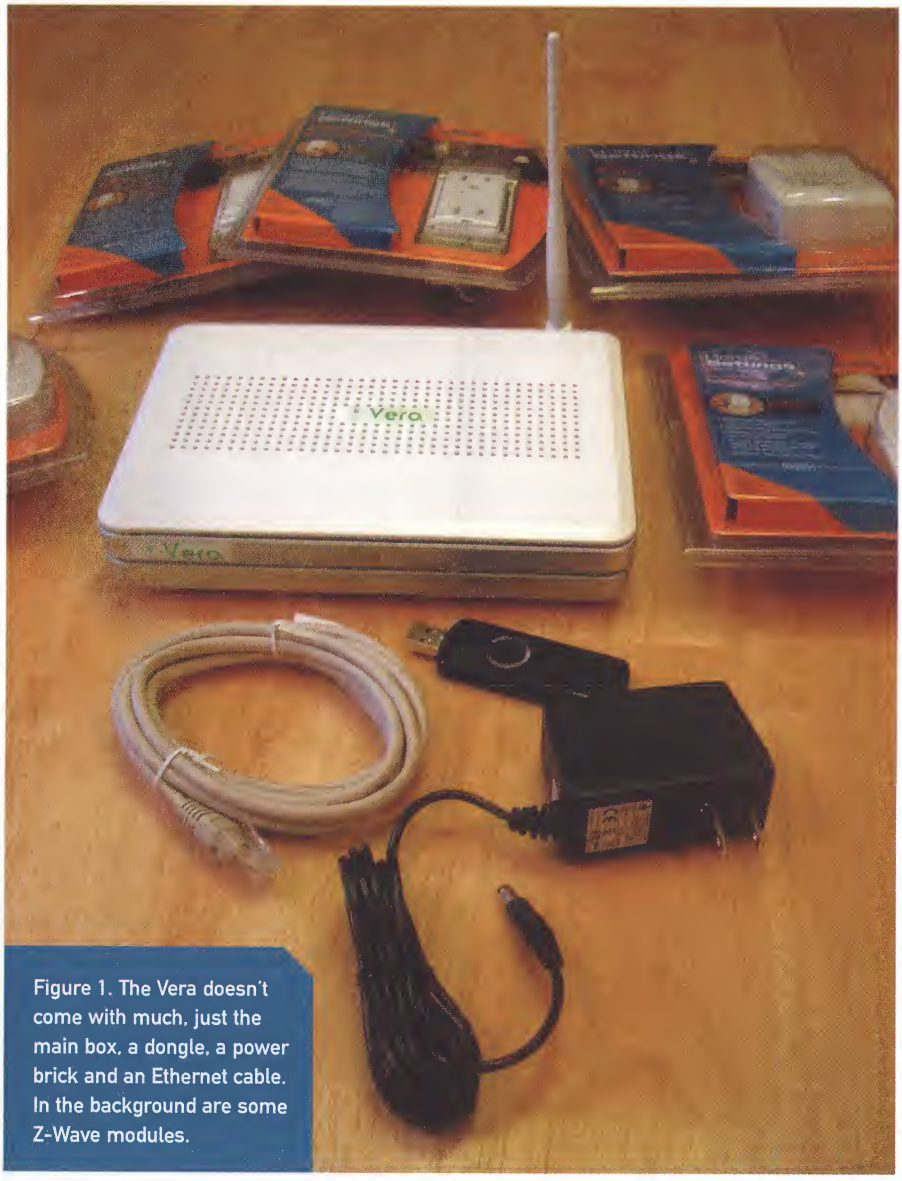


Figure 1. The Vera doesn't come with much, just the main box, a dongle, a power brick and an Ethernet cable. In the background are some Z-Wave modules.

Automating your home is one of those ideas that has been around forever. The idea is older than the light bulb, which was itself a form of automation—no more lighting gas lamps or candles manually, simply flip a switch.

We humans are a lazy bunch—even a simple action like flipping a switch became not easy enough—so various devices and technologies have appeared over the years to automate lights and other electrical devices. The type of devices available to control your house basically can be split into two categories: professionally installed and do-it-yourself.

For all home-control systems, there has to be a way to tell devices to turn on and off. For professionally installed systems, this often means the installation of extensive new wiring, which is fairly easy to put into place if you are building a new house but can be very expensive if you are trying to install a system into an existing house.

For the do-it-yourself crowd, the most popular system has been X10, which sends signals over power lines. This has the advantage of not needing any new wiring. You can build your system out slowly, and you can do it yourself as long as you are reasonably technical. However, X10 has several disadvantages. If you live in an apartment, for example, and you and a neighbor are both using X10, the potential exists for you to turn each other's lights on and off accidentally. Even in a house, things might not work as expected, because sending signals over power lines is problematic, and sometimes interference causes signals to be lost or misdirected.

Z-Wave is a new automation technology that attempts to overcome the limitations of earlier systems by utilizing wireless mesh networking. Each Z-Wave device acts as both a transmitter and a receiver. Any signals it receives are retransmitted automatically. Additionally, whenever it acts on a command, a Z-Wave device sends out a reply message letting the controller know that it acted.

Like X10, Z-Wave lets you automate your home a few devices at a time, which allows you to spread the cost

Z-Wave is a new automation technology that attempts to overcome the limitations of earlier systems by utilizing wireless mesh networking.

over a longer period, rather than having to pony up a lot of cash up front. There's nothing stopping you from doing it all at once or even from hiring a professional to do it for you, but you don't have to.

The main problem that Z-Wave has, which is common to all home-control systems, is it can be complicated to set up.

One new Linux-powered product, the Vera from Mi Casa Verde, advertises itself as a Z-Wave home-control gateway that anyone can set up. To test that claim, I gave an unopened Vera and several Z-Wave modules to a nontechnical test user, and asked the user to set it up.

The Vera comes with a four-page printed setup guide that walks you through plugging the Vera in and connecting to it for the first time. When it is first turned on, the Vera tries to connect to your home network using DHCP. As the test user knows next to nothing about switches, routers, Ethernet or DHCP, and never has needed to, I had to assist with this. I helped the user plug the Vera in to a free spot on a switch, and the user powered it on.

The documentation at this point states that you should wait a couple minutes after turning the Vera on so that it can boot and announce its presence to the findvera.com Web site. After waiting, all you have to do is go to **findvera.com** and click on the big green Setup Vera on my home network button. The user did this and was able to connect to the Vera Web interface.

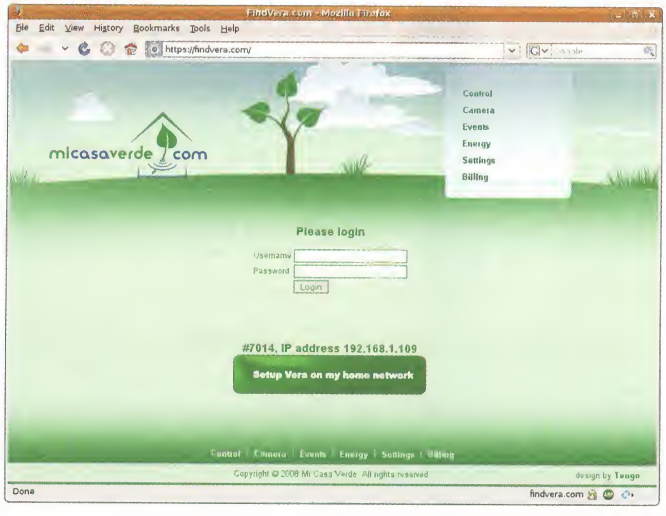


Figure 2. When connecting to Vera, you don't have to know the IP address, simply go to findvera.com and it will connect you.

So far so good. The Vera was plugged in, and the user was able to connect to the Vera without knowing what address it had been assigned by my home router. This integration is slick and makes connecting easy.

The next step in the printed documentation has a screenshot of what you should see after you connect to Vera for the first time. However, when the user connected the first time, the Vera went to the firmware upgrade screen instead of the initial setup screen. The user found this confusing. The printed documentation should state that the firmware upgrade screen might appear and what to do about it before continuing with the initial setup. Or, the Vera should have realized that it was not set up and waited until it was before prompting to upgrade the firmware.

Upgrading firmware on the Vera is easy, and it prompts you to make a backup of your settings before starting, which is a nice touch. However, it is not a process the user was comfortable doing, so I finished that step before we moved on.

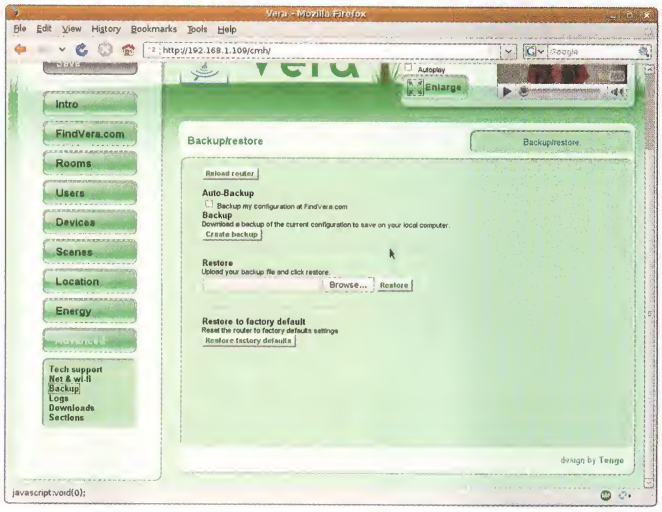


Figure 3. Backing up the Vera configuration is simple. Restoring also is easy.

After updating the firmware and rebooting the Vera, the user connected again, and this time, instead of seeing the setup screen, the Vera went to the Dashboard. This screen is supposed to appear automatically after setup, and it provides a single screen from which to control your devices. It is not what the user expected to see, so the user wasn't quite sure what to do since the page was basically blank (nothing had been set up yet at this point). I directed the user to the setup link at the top right of the screen and after clicking on it, the screen appeared that the printed documentation said you should see.

At this point, the printed documentation basically ends (apart from some network troubleshooting information), and the Vera relies on embedded YouTube videos to talk you through what to do next and how to set things up. These videos play automatically by default, which the user found useful. This autoplay feature can be turned off by unchecking the Autoplay button.

The basic sequence of events for this initial setup is to walk your way from top to bottom through the buttons on the left-hand side of the Vera Web interface. The first step (apart from the Intro section) the Vera asks you to complete is to set up an account on findvera.com, so you can connect to your Vera

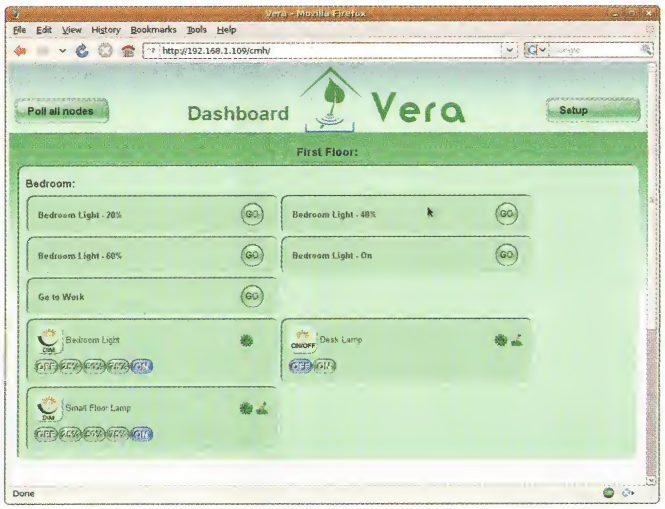


Figure 4. The Dashboard serves up all of your devices on a single page for easy control.

Figure 5. Helpful videos play automatically to walk you through the initial setup of the Vera.

from anywhere in the world. The use of this remote-access service is free for the first 90 days and costs $7.95/month after that. The on-line documentation does state that the service is optional, and it even points out that you can set up external access yourself if you want to. The Web GUI doesn't mention any of this (you have to click on a "learn more" link to get the information), so I just told the user to skip to the next section.

The next step, and the first real one in my opinion, is to create "Rooms". This is so the Vera can organize your devices logically. The user found this step easy and did not need any help from me. The embedded video on this page even helpfully suggests that if you have outdoor devices, you should just think of them as rooms to keep things simple.

The next step is to add actual Z-Wave devices to the Vera. Devices range from simple lamp and appliance modules (lamp modules are dimmable, and appliance modules are not), to motion detectors, automatic blinds and thermostats. Lamp and

Advertiser Index

CHECK OUT OUR NEW BUYER'S GUIDE ON-LINE.
Go to www.linuxjournal.com/buyersguide where you can learn more about our advertisers or link directly to their Web sites.

Thank you as always for supporting our advertisers by buying their products!

appliance modules come in several varieties, from small boxes that you plug in to wall outlets, to actual in-wall outlets and light switches that you install in place of the originals. Obviously, the in-wall devices will give your room a cleaner, more professional look, but they are harder to install. The Vera also can interface with some non-Z-Wave devices, including IP cameras.

The basic sequence for adding new devices is as follows:

1. Unplug the Z-Wave dongle from the back of the Vera.
2. Take the dongle over to the Z-Wave module that you want to add.
3. Press the button on the dongle (it will start to blink).
4. Press the button on the Z-Wave module (the dongle light will go solid for a couple seconds to let you know it detected the button press).
5. Repeat steps 2–4 for any other modules you want to add.
6. Plug the dongle back in to the Vera.

The steps were easy to understand for me, but they were not as easy for the user. The confusion mainly centered around the terminology. For example, the user was not familiar with the word dongle and wouldn't have known what it was if the documentation hadn't included a picture and mentioned that it was black. The user said the word dongle was about as descriptive as the words thingy and whatchamacallit. Even with this bit of confusion, the user was able to add the devices without help, thanks to the clear instructions.

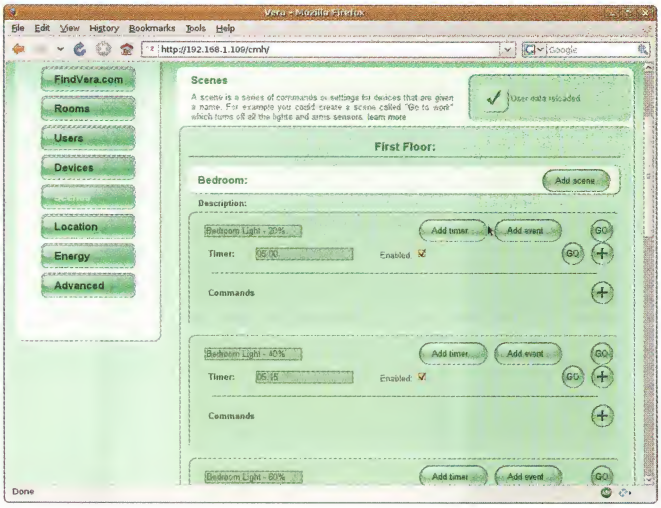


Figure 6. The GUI for configuring scenes has a confusing button layout.

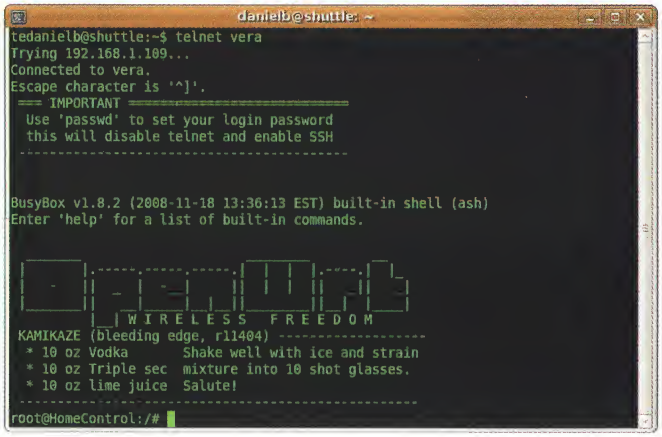


Figure 7. You can Telnet to the Vera to get into the guts of the device.

Once the user added the devices and plugged the dongle back in to the Vera, it was time to configure them. The user also found this difficult the first time and asked for my help on the initial one.

You configure devices by setting up various "scenes". Scenes are basically groups of commands. An example would be "set all lights in the family room to 50% brightness", or "turn all lights off in the master bedroom". Once you have set up a scene, you can call it quits, leave it as is and run it manually at any time. However, the real power of the Vera comes when you add timers and events to scenes.

Timers let you run scenes at specified times, such as "every Monday at 8pm", "every 2 hours" or "on 12 June 2009 at 3:24am". Events let you run scenes when a certain event occurs, such as "whenever the doorbell rings", "if the hall motion sensor is tripped" or "when the master bedroom light switch is turned on". Once I walked through the process of

"Velocity fills a very important hole in the art of building great on-line services: a focus on performance that no other conference can match. The 2009 Velocity conference looks to be the best yet, with sessions that cover all layers of the web application stack."

—Jeremy Zawodny, Craigslist

O'REILLY

Velocity

Web Performance and Operations Conference

JUNE 22–24 | 2009

SAN JOSE | CA

conferences.oreilly.com/velocity

<FAST> <SCALABLE>
<EFFICIENT> <AVAILABLE>

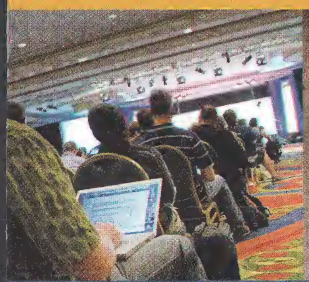
Building a Better Internet

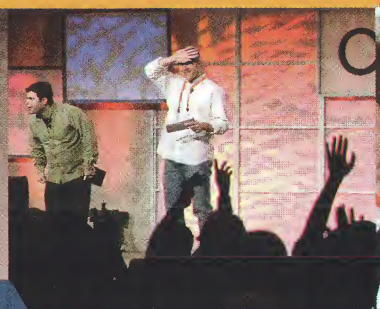
Velocity, the Web Performance and Operations conference from O'Reilly, is the place to be in 2009 for crucial skills and knowledge that will help you build web sites that are fast, scalable, resilient, and highly available.

Featured Speakers:

- **Bill Scott,** Netflix
- **Jonathan Heiliger,** Facebook
- **Douglas Crockford,** Yahoo!
- **Mandi Walls,** AOL
- **Nicholas Zakas,** Yahoo!
- **John Allspaw,** Flickr
- **Jeremy Zawodny,** Craigslist
- **John Adams,** Twitter
- **Steve Souders,** Google

REGISTER WITH CODE VELO9LJG AND SAVE AN EXTRA 15%!






After the Vera decided to turn on the lights in the room at 5:00am, I had had enough, and I removed the glitchy event.

setting up an event and a timer, the user understood it and was able to add scenes, timers and events.

You probably are noticing a pattern here. At nearly every step, the user got stuck and came to me with questions. One thing the Mi Casa Verde folks could do to alleviate at least some of this would be to show someone actually completing the steps instead of just telling you how to do it. Perhaps they also could have you configure a virtual room with virtual devices that you can configure and play with. The fact that they are attempting to simplify things to the point where anyone can automate a home is a lofty goal, and I think they actually have succeeded in many ways, even though I ended up doing most of the configuration.

The basic flaw is that Vera assumes you know what certain keywords mean. You are expected to know what dongle, LAN, Wi-Fi, DHCP, gateway, USB port and other terms mean. For a reasonably technical person, those terms are easy, but for someone like the nontechnical user in this example, such terms might not be understood. For this user, a computer is just there—you sit in front of it, type and click the mouse. The concepts of files, applications and programs are needlessly complicated. To this user, there are only tasks, such as "check my e-mail to see if Sue has written back", "print this letter so I can mail it", "watch a video on YouTube", "see if check #1234 has cleared the bank", "upload photos from the party last night to Facebook" and so on. The user has no desire to move past that level of understanding. Vera and other products targeted at everyone need to realize that even the phrase "unplug the dongle" might not be understood.

Lights

Being open source, the Vera is quite hackable. One thing you can do is send it special HTTP queries to control lights. Below is a simple bash script I wrote to turn all of the Z-Wave devices in my house on or off. It's not the most elegant script in the world but it works:

```
#!/bin/bash
# This file is named "lights" and is placed in
# /usr/local/bin with chmod 755

# lights in the house
#   biglamp="12"
#   smalllamp="13"
#   desklamp="14"
#   tv="16"
#   masterbedroom="17"

# All of the lights in the house
lights="12 13 14 16 17"

function turnlight() {
    if [ "${2}" = "on" ]; then
        # Turn the light on
        curl \
          "http://vera:3451/messagesend?from=1&to=${1}&type=1&id=192"
    else
        # Turn the light off
        curl \
          "http://vera:3451/messagesend?from=1&to=${1}&type=1&id=193"
    fi
}

if [ "${1}" = "on" ] || [ "${1}" = "off" ]; then
    for light in ${lights}; do
        turnlight ${light} ${1}
    done
else
    echo "Usage:"
    echo "   \"${0} on\" to turn all lights on"
    echo "   \"${0} off\" to turn all lights off"
    exit
fi

exit 0
```

Issues

I found the Vera quite easy to set up and use overall. However, I did run into several issues.

The Web interface is incomplete. Some sections do not have an explanatory video or any other documentation. I'm sure documentation is coming, but that doesn't help me right now.

Besides missing bits, the Web interface is also glitchy. It relies on AJAX-style automatic form submission when you add devices, scenes, events and so on. This works fine most of the time, but I ran into problems a couple times where configuration changes I made were not applied, and I had to enter them a second or even third time before they "took". The explanatory videos also stopped during playback several times while the interface performed housekeeping on something or other.

Pricing Information

The Vera is $299 from the Mi Casa Verde on-line shop.

Z-Wave modules start at around $35 and go up from there, depending on the brand and features.

I purchased my Z-Wave modules from Amazon.com—simply search for "z-wave" for a list of the available modules. I purchased the following:

- Intermatic HA06C Wireless Indoor Wall Switch: $36.86
- Intermatic HA03C Wireless Plug-In Indoor Lamp Module: $32.54
- Intermatic HA02C Wireless Heavy-Duty Plug-In Appliance Module: $39.97
- Intermatic HA01C Wireless Wall Receptacle: $33.79

Also, on at least two occasions during my testing, the Vera stopped working altogether. During these incidents, the Web interface still was responsive, and it acted like things were working, but none of the lights would turn on or off when told to. I still could turn them on and off via the buttons on the individual modules or by using my Z-Wave remote. A reboot of the Vera solved the issue. I do appreciate the improvements the firmware upgrades have provided, but I hope stability and reliability are at the top of the list for the Vera developers, especially as I continue to add more modules.

Finally, I ran into issues with the Vera doing crazy things on me. I tried at one point to set up an event that would turn on my bedside lamp whenever the ceiling light was turned on. After setting it up, the Vera started doing strange things like dimming and then brightening the lights in the room, shutting the lights off at random times, turning the lights on at equally random times and so on. After the Vera decided to turn on the lights in the room at 5:00am, I had had enough, and I removed the glitchy event.

Conclusion

Thankfully, none of my other timers and events have been as troublesome as the bedroom one.

In fact, the Vera has been very reliable about most of my scenes. I have one that momentarily dims the lights in the family room when it is time for the kids to get ready for bed. I'm still working on the part where the kids actually start getting ready for bed at that point, but the scene itself works flawlessly. All of my other scenes, events and timers also have worked well.

Above all, the thing that Vera does do well is hide a lot of the complexity of setting up and operating a home-control system, even if it didn't quite pass the non-geek user test this time around. ■

Daniel Bartholomew lives in North Carolina with his wife and children. He can be found on-line at daniel-bartholomew.com.

Resources

Mi Casa Verde: **micasaverde.com**

Mi Casa Verde Forums: **forum.micasaverde.com**

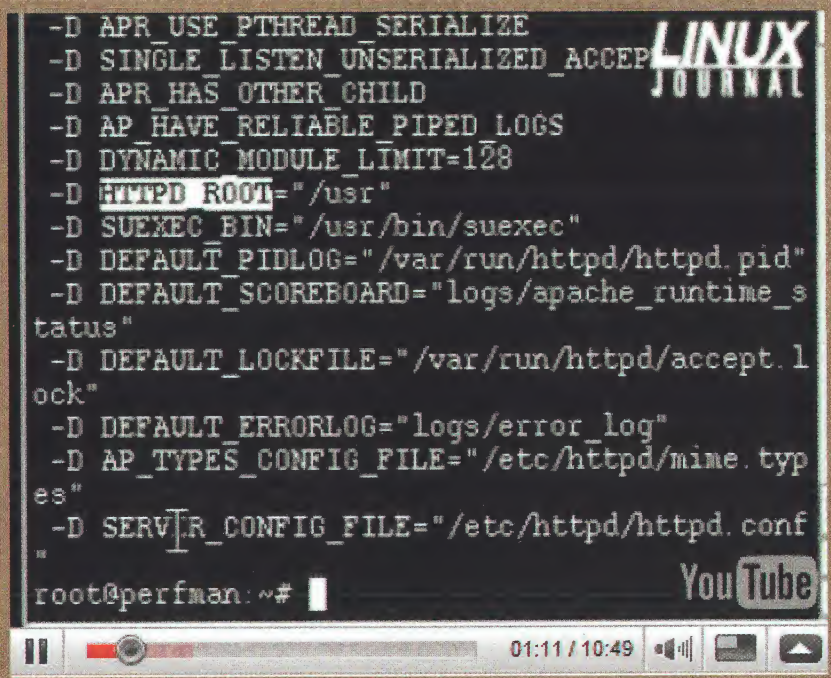
Mi Casa Verde Wiki: **wiki.micasaverde.com**

Mi Casa Verde On-line Store: **https://shop.micasaverde.com**

Amazon has a good selection of Z-Wave devices: **amazon.com**

ON THE WEB, ARTICLES TALK!

Setting up an https server in Apache is easy. This tutorial covers how to create and sign your ssl certificate as well as how to configure the Web server: www.linuxjournal.com/video/set-secure-virtual-host-apache.



Interview with Joe Born: CEO of Neuros Technology

Joe Born talks about his company's Neuros products and how open devices are upending the consumer electronics industry like never before.

JAMES GRAY

The consumer electronics (CE) industry is in an upheaval as devices become more open and a rift emerges between hardware manufacturing and the software that steers them. On the cutting edge of this development is Neuros Technology, which has brought the Linux and open-source model to CE devices for TV-Internet convergence. I recently spoke with Joe Born, Neuros Technology CEO and founder, to learn more about Neuros Technology and where this exciting trend toward open CE devices is headed.

Joe Born, CEO of Neuros Technology

JG: First, thanks for joining us in this conversation, Joe Born. The open devices from your company, Neuros Technology, make perfect sense to us Linux and open-source geeks but are quite disruptive in the world of electronics. Can you start us off by giving us a quick rundown of your products and how they are different from your typical set-top box?

JB: Well, they are open. Now, normally that's associated with open source, but actually electronics devices today are vastly more closed than any Windows PC dreamed of being. If you look at pretty much *all* the electronics devices that power the TV, it's not just that they don't allow modification, they don't even allow you to browse outside the walled garden that they have set up. Compared to a PC, they are closed at every level.

So in the Neuros LINK [device], that means it can browse to any site, and you get access to all the content you can find, compared to just about any other set-top device you can imagine—from the operator boxes to the AppleTV to TiVo and so on.

Basically, Neuros is looking to create a device that fills the gap between the typical electronics that connect the Net and TV (of varying shapes and sizes) and the wide-open HTPC. We want to provide navigation and ease of use like a CE device, but with the openness of the PC. Enabling that functionality is a host of free software. Under the hood, the LINK is really a diskless, quiet PC, and with all the power and expansion of the PC, but over time, we're adding all the seamless navigation of a nice electronics device.

As to our other products, Neuros TV is what a TVPC should be: an open device that can stream virtually any Internet content to your TV. Building on the lessons learned from the closed, proprietary solutions, Neuros has built a device that's quiet, component-sized and sets up easily with all the peripherals you need and none that you don't. It's different from your typical set-top box solution, because it allows you to access any content of your choosing easily, not just the one your provider or manufacturer decides you should have.

Then, there is the Neuros OSD, a standalone device for archiving all your DVDs, VHS tapes and TV shows into unlocked digital recordings. It's particularly good for making recordings that play on your handhelds (iPhone, Android and so on) with no hassle or conversion.

JG: In a nutshell, what is the story behind your company, and where did the inspiration come from to create open devices?

JB: Like most manufacturers, Neuros didn't pay a lot of attention to the Open Source movement initially, viewing the various activities on Linux, Web servers and browsers as an interesting but distant phenomenon with little obvious connection to our business. It was almost by happenstance that we realized what a profound impact on our business the open-source phenomenon could have.

Our first surprise came even before we released any source code. We had simply released the communication protocol between our device and the PC. Based on that small release of information, brand-new synchronization managers sprang up, as if from thin air. They typically were developed by engineers who often had no contact with the company. The software was innovative and took entirely different approaches, and in many cases, it was preferred by many users to the software we had spent literally millions on developing in house. Equally amazing were the tools they were using. These independent, open-source developers had complete toolkits of free software, that were, in many cases, vastly superior to the proprietary ones we had been using.

I can remember the first time I saw the bug-tracking software called Bugzilla that many of the open-source developers were using. Like most companies, we had purchased proprietary software to track our software bugs and enhancements and communicate updates throughout the company. I remember being amazed when I first saw Bugzilla. Not only was it free, but it had all kinds of features we'd long been looking for. Not only that, but its open-source license meant we could put it on a public server

that anyone could access. Suddenly, we had the ability to tap directly into our most sophisticated and enthusiastic users for finding bugs and, even better, making suggestions and enhancements. Not only that, but Bugzilla had a voting function that meant the public could chime in on its priorities. Overnight, our consumer intimacy would jump five-fold with this ability, I thought. When I asked our internal team, no one could think of a single reason we should stick with our old proprietary closed system.

Although everyone agreed that Bugzilla was a superior system for tracking bugs, there were, however, plenty of concerns about exposing our internal bug-tracking system to the public, particularly from the marketing department. Would users be turned off by being able to see our list of bugs? Would we be able to control a system where hundreds of unscreened users have access to input and comment on all the bugs? Would users be offended when we decided not to make an enhancement they had suggested?

In the end, we decided it was worth the risks, and that as Bugzilla was capable of supporting a public-based system, why not use that functionality? In the years since we made the system public, none of our fears have come true. In fact, there has been substantially less complaining by users, perhaps because we have given them a constructive outlet to report their issues. Further, our connection with our customers has increased dramatically, and we now have a systematic way to include their input into our internal plans. This level of consumer input could never be duplicated with conventional market research. To date, the concerns about an open system spiraling out of control have turned out to be unfounded as well. As quickly as duplicate or irrelevant bugs are entered, they are corrected, as the community effectively polices itself.

Perhaps not surprising to those experienced in open-source development, our introduction to open source as users of the software quickly led to our embrace of open source as a development method—a method that, with heavy doses of experimentation, mis-steps and modifications, we could apply to hardware development as well as software.

JG: Does "the industry" understand what you guys are up to, or is it too myopic to really get it?

JB: This is an incredibly rich area for discussion, and it really depends on what you mean by "industry" and "get it".

From our viewpoint, it's plain to see that the electronics industry is undergoing a change that very much mirrors the PC industry 25+ years ago. Devices are undergoing a transformation from being dedicated, closed devices to more open ones, mirroring what happened going from what were essentially word processors to the IBM PC in the early 1980s. Today, the silicon behind electronics has become powerful enough that it has outstripped the ability of the folks manufacturing it to create the software for it, and a natural separation between hardware and software has emerged. This is further splitting into operating systems, applications and services, and we can see some of this already happening on the iPhone, for example.

Nowadays, much of not just the manufacturing but the design work also is being done in Asia. Certainly these "design manufacturers" or ODMs, get this separation very well. They know that software teams have to be close to the customers, and they recognize that they have neither the resources nor the expertise to develop the applications that are necessary to make devices successful today. One interesting area is really in the operating system, actually. The free nature of Linux is leaving the operating system a bit "up for grabs", so to speak. There are vendors, like MontaVista that do a nice job with this, but their business models can't be the same as Microsoft's (or Apple's for that matter). My personal gut feeling is that branded distributions will emerge here, and things like mobile Ubuntu, Android and maybe something coming from Nokia will emerge. The silicon manufacturers have a strong role to play here in providing turnkey solutions for their customers. TI is leading the way with some of its efforts (both on its own and supporting our efforts), and lately we've begun working with ATI in improving support for Linux.

I see a lot of activity on the supply chain per the above, so I would generally say they get it. Downstream on the brand side, there's a bit more resistance.

The branded manufacturers are a bit more of a mixed bag. There are reasonable reasons for this.

First, consumer expectations are different for an electronics device than a computer. Consumers are not used to an open system, so their expectations are for more of a controlled experience.

Second, there's still a bit of old-fashioned thinking and fear about opening a system and losing control.

Third, there are impediments given the DRM and encryption issues.

And fourth, the "customers"—that is, cable and wireless operators—often don't want open systems, because they want to control their networks and devices (for a variety of reasons).

So, on the branding end, there's more resistance, but that's where Neuros comes in. I believe we can pioneer this area and demonstrate enough consumer demand that it will overcome the above.

JG: You've used the term "super ODM" to describe Neuros. What does that mean?

JB: An ODM is a Original Design Manufacturer—basically a factory that adds design capability and initiates development of products, providing a more turnkey solution to its customers. Neuros considers itself a "super ODM", meaning we not only take responsibility for design, but also launch our products directly to involved users, get immediate feedback from those customers and evolve the product. This means that, to our customers (the larger electronics brands), we have not only done design work, but also evolved the products in direct response to users' needs, proven the early market for the devices and taken a lot of the risk out of the process for our customers.

JG: How are you collaborating with other companies committed to open devices and open source?

JB: "Supply chains" of both software and hardware are long and segmented today. There are so many different contributors and pieces to any piece of electronics equipment you buy today, it's just inherent that you are borrowing from the contributions of many, many levels. One of the things that makes open source so compelling is that it forms this giant ecosystem. Without a single business development meeting, without a single nickel in legal fees, we have a mature agreement in place (typically the GPL) with a huge ecosystem of projects and companies; these are crucial building blocks for getting out products quickly. The addition of more and more commercial entities to that ecosystem is a huge boon. We're in discussions [for example] with Boxee as we speak, figuring out how best to deliver a product that incorporates their software and services.

JG: I bet it's really fun to have such a dynamic group of contributors outside the walls of your office.

JB: It's incredibly gratifying, and I've come to realize what a special thing it really is. It's much more than a bunch of smart folks working together in a community.

JG: In a separate conversation, you told me the fascinating story one of your most prolific contributors. Could you share his story with our readers?

JB: Pablo Grande was the most prolific and talented hacker in the Neuros community. He contributed at every level—from low-level assembly language hacking all the way to setting up the community Web sites. But amazingly, his greatest contributions were made after he had a severe stroke in 2005. After the stroke, Pablo's heroic recovery and participation not only inspired but demonstrated to the Neuros community the power of open development. We all came to realize that what we had considered a nice little on-line community was really something more—open communities were creating a place that, as melodramatic as it sounds, was really unlocking the power of human potential. Here was Pablo, without the use of one hand and, at the time, unable to speak, still able to bring his exhaustive knowledge and insight to bear on the problems facing Neuros. Where else could he contribute at that level? Where else could he prove what he was really capable of, that probably only he could see? What other type of institution would be able to accept contributions solely limited by the contributor's own ability? We realized watching Pablo's example that at various levels it was true for all of us. Unlike typical, top-down corporate development, we were all contributing in a way that really was limited only by our own energy and ability. Since that realization, I know I personally have felt a passion for open development and what it can do, not just for the projects themselves but for the contributors as individuals.

JG: What are some of the most innovative contributions you've received from contributors to the Neuros OSD?

JB: Well, our YouTube browser was one open-source contribution, then an audio player, and then a third contribution was a mashup that stitched those two together allowing you to browse MP3s, and then with a single click, "find the music video" for the song. I thought it was a neat project, but what I really liked was the cooperation between community members.

Recently, we really had a lot of fun with "crowd narration", a technology that superimposes two lines of chat text over a video broadcast, effectively allowing individuals to provide commentary in real time to live events or shows. It's a kind of closed captioning for crowds.

[For a video illustrating crowd narration, see **open.neurostechnology.com/content/crowd-narration-future-tv**.]

Honestly though, I think the fun has just begun. We've seen more interesting experimentation since we've launched the LINK than in all the previous history combined, and the reason is simple. The product is further along. By using an x86 processor and Ubuntu, we made experimentation and enhancements more accessible. Now, unlike in the past, the first 95% is already done. Basic functionality already works, and now it's about the really interesting stuff of presentation, sharing, discovery of good content, interactivity and so on.

JG: Do you see a conflict between the needs of users of your devices and the developers who contribute to them? If so, how do you mitigate this conflict?

JB: Sometimes, there's a conflict. Ease of use and "intuitive" is certainly defined differently for developers and mainstream users. The wisdom of the crowds isn't, and never will be, a substitute for individual judgment and leadership. We still ultimately have to make the call on things like those conflicts.

JG: Your development team holds its developer meetings on public IRC. How has this experience been, and is the goal of transparency well served in this way?

JB: It has worked well. As you probably know, a community isn't a monolithic block; there are folks who are very close to the core and their "influence" radiates outward to the less-involved users and customers. By keeping the most involved contributors in the fold, they can spread the word in their own ways and make sure it gets out to everyone. If you look at our forums for example, many of the folks answering questions are not internal staff. Having open meetings is a good mechanism for making sure that those folks, as an example, understand well what's going on and can articulate it to others. Without transparency, this would be impossible.

JG: What do you think will be the significance of Neuros LINK and Neuros TV in the marketplace?

JB: Well, see above. But, in broad terms, I think opening up the TV will have more impact on worldwide freedom of communication than the creation of the PC. There are vastly more TVs in the world than PCs, and you'll find them in some of the areas where freedom of communication is most limited. Once cheap hardware connecting those devices in a decentralized way penetrates those areas, the implications are hard to fully imagine at this point.

JG: Does Neuros Technology have plans on the horizon for other devices, products or services?

JB: Well, we do, but I believe we'll be in this category for a while. I expect there will be integration with PVR functionality and perhaps other types of functionality, and of course, continuous improvement within this category. There's lots to do.

JG: Thanks, Joe Born, for sharing your insights on Neuros Technology and open devices.■

James Gray is *Linux Journal* Products Editor and a graduate student in environmental sciences and management at Michigan State University. A Linux enthusiast since the mid-1990s, he currently resides in Lansing, Michigan, with his wife and cats.

Resources

Neuros Technology Home Page: **www.neurostechnology.com**

About Unlocked Media: **open.neurostechnology.com/content/unlocked-media**

LJ Video Review of the Neuros OSD: **www.linuxjournal.com/video/review-neuros-osd**

"The Neuros OSD Connects Your TV to the Internet" by Marco Fioretti, *LJ*, August 2008: **www.linuxjournal.com/article/9999**

LJ Video Review of the Neuros MPEG4 Recorder: **www.linuxjournal.com/video/review-neuros-mpeg4-recorder** [Editor's note: this is an antiquated product that is no longer available from Neuros.]

OpenFiler: an Open-Source Network Storage Appliance

Turn that old computer into a network appliance with OpenFiler, an open-source alternative to a NetApp filer.

BILL CHILDERS

I've set up quite a few file servers using Linux in my day, and although it's not particularly difficult, I've often thought that there should be a better way to do it. The folks at the OpenFiler Project definitely have built a better mousetrap. The OpenFiler team seems to be inspired by the NetApp filer family of Network Storage Appliances and has come out with an open-source clone that lets you take any x86 computer and give it nearly all the functionality of a NetApp filer.

About OpenFiler

The OpenFiler distribution is an easy-to-install, easy-to-use, nearly turnkey solution. At the time of this writing, the current version is 2.3, and it's based on rPath, so it's focused and lean where it needs to be, allowing the developers to pack it with features useful to its main purpose. It's even lean enough to run on some embedded systems. The feature list is comprehensive, and it compares very well with commercial appliances like those offered by Snap and others. Here are some of OpenFiler's killer features:

- Full iSCSI target and initiator support.
- Support for Fiber Channel devices (depending on hardware).
- Support for software (md) RAID or hardware RAID.
- On-line volume/filesystem expansion.
- Point-in-time snapshots.
- Synchronous/asynchronous replication of data.
- NFS, SMB/CIFS, HTTP/WebDAV and FTP.
- Supports SMB/CIFS shadow copy for snapshot volumes.
- Supports NIS, LDAP and Windows NT/Active Directory authentication.
- Flexible quota management.
- Easy-to-use Web-based admin GUI.

The only real downside to OpenFiler is that you have to pay for the Administration Guide. The Installation Guide and a downrev version of the Admin Guide are both on-line and available for free, but the current revision of the Admin Guide is available only for paying customers, as this is how the OpenFiler Project is funded. Luckily, OpenFiler is easy to configure, thanks to its GUI, so that isn't a huge detriment.

Installing OpenFiler

If you are familiar with installing a Red Hat-based Linux distribution, installing OpenFiler will be old hat to you. The system requirements are fairly low. I've installed OpenFiler on an embedded PC with a 500MHz CPU, 512MB of RAM and a 2GB CompactFlash in this case, but it'll install on regular desktops and servers as well. Booting off the CD lands you into a graphical installer (unless you use the text argument when booting the system). Note that you must select manual partitioning when setting up the operating system disk in your machine; otherwise, you won't be able to set up data storage disks in the OpenFiler Admin GUI later. Aside from that, it's a fairly standard Red Hat-ish installation. Once the installation is complete, the next step is to configure your OpenFiler instance by pointing a Web browser to https://IP_OF_OPENFILER:446.

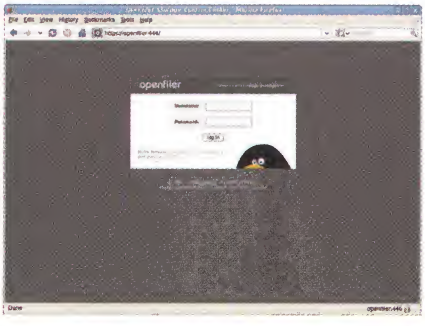


Figure 1. Logging In to OpenFiler

Configuring OpenFiler

You now should have the OpenFiler management GUI open in your Web browser, as shown in Figure 1. As per the Installation Guide, log in with user name "openfiler" and password "password". After you log in, you'll be in the admin interface, at the main status screen. From here, you can configure just about every aspect of your OpenFiler.

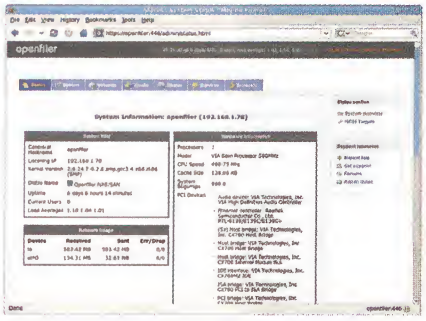


Figure 2. Admin Console: Status Screen

The status screen can show you vital system information at a glance. It's especially handy that the admin interface displays the uptime and load average of the machine in the title bar of the console. Not shown in the screenshot are the memory and storage graphs, similar to a graphical top.

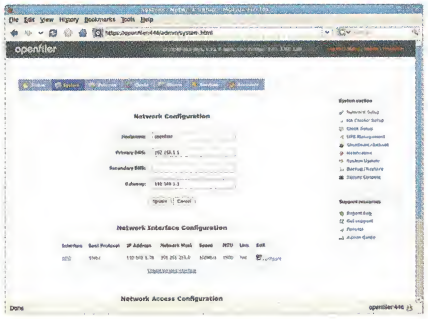


Figure 3. Admin Console: System Screen

The system screen is where you can set up and adjust the overall system parameters, like the IP address of the machine or its high-availability/replication partner. It even embeds a Java-based SSH client in the console, so you can get a shell on the machine if you need to, although any SSH client works as well. Note: it's important to define the hosts or networks that your OpenFiler will serve here. If you don't do that, your OpenFiler will refuse to serve files via NFS or SMB/CIFS. It's not difficult to add—I simply dropped a statement to cover my 192.168.1.0/24 in there—but OpenFiler stubbornly refused to talk to any machines until that was added. Another thing to note here is that OpenFiler supports the creation of bonded Ethernet interfaces, so if you're building a mission-critical file server, you can put two network cards in the server, connect each card to a different network switch, and then you have fault tolerance at the network level.

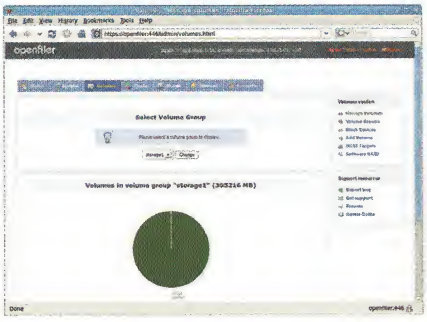


Figure 4. Admin Console: Volume Manager

The volume manager is where you can add disks to your OpenFiler, create filesystems and manage software RAIDs. OpenFiler uses the Linux Logical Volume Manager (LVM) as its volume manager, and it supports both ext3 and XFS filesystems for storage that's locally attached to the OpenFiler host. In this case, because I'm using an embedded PC, I had to attach a 320GB disk via USB to OpenFiler. It wasn't a problem—OpenFiler happily allowed me to create a volume group using that USB disk, and then I could create a volume within that group and start laying out the filesystem.

The next tab in the admin interface is the quota tab. The quota screen lets you set quotas per group, user or guest, and have a different quota for each volume. For example, if your OpenFiler was in a business environment, you could set everyone in the Marketing group to have a 2GB quota each, everyone in the Engineering group could have a 10GB quota, and everyone in the IT group could be uncapped—except for the CEO, who's also uncapped. Having flexible quota options allows you to tailor the OpenFiler to the needs of your business.

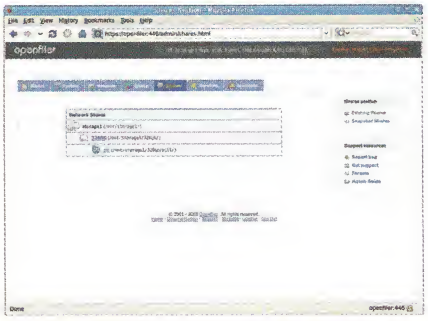


Figure 5. Admin Console: Share Manager

The share manager is where you make subdirectories within a volume, and then share out those subdirectories. This is where you'll spend a lot of time, setting up the directories, shares and access permissions. A nice feature of OpenFiler is that you can specify which network service shares out a specific directory. For example, I can set up a Sales share that is SMB/CIFS only (all the Sales folks run Windows), an Engineering share that is NFS only (all the Engineers run Linux) and a Sandbox share that is serviced by both SMB/CIFS and NFS. I then can use the same screen to lock down the permissions on the respective shares, so that only the members of those groups can read or write to those shares, while the Sandbox is wide open.

I discovered an interesting bit of trivia while researching this article. If you want to share directories via NFS to an Apple Mac, so the directory can be mounted in the Finder, you must specify that the share's origin port be above 1024 (this is otherwise known as an insecure NFS option). The Mac won't talk to NFS servers running on privileged ports. (And yes, I have a Mac. I view it as a flashier but less knowledgeable cousin to my Ubuntu machines.)

The next tab over is the services manager, where you can enable or disable the network services provided by OpenFiler. If you plan on using your OpenFiler only as an NFS server, you can turn off the SMB/CIFS services completely and save some memory on your server. This screen also is where you can specify options, such as of which workgroup the SMB/CIFS server is a member or whether there is a UPS attached to the OpenFiler, so it can auto-shutdown in the event of a power failure. OpenFiler also can act as an LDAP server, and you can back up or restore LDAP directories via this screen.

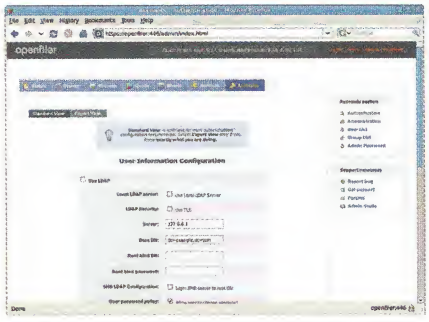


Figure 6. Admin Console: Accounts Manager

The last tab in the admin console is the accounts manager, which is where you define what authentication methods you'd like OpenFiler to use. You can run an internal LDAP server on the OpenFiler itself, and create the users and groups locally. You also can point the OpenFiler to your corporate LDAP if you have one. If you're in a Windows environment, you can set up OpenFiler to use your corporate Active Directory for authentication or even an old-school NT4-style domain.

Under the Hood

Underneath the GUI interface, OpenFiler is powered by a bunch of open-source software. At its core, it is an rPath OS

Installing OpenFiler via PXE

The little embedded PC on which I installed OpenFiler doesn't have an optical drive, so I had to install the distribution via PXE. I copied the distribution CD to an NFS server and exported that directory via NFS. Then I copied the vmlinuz kernel file and initrd.img initrd archive from the /isolinux directory on the CD to the tftp directory on my PXE server. The last step was to add the following lines to my PXE server's pxelinux config:

```
LABEL openfiler
KERNEL vmlinuz
APPEND initrd=initrd.img text askmethod ramdisk_size=8192 console=tty0
```

After doing that, installing OpenFiler was as easy as booting my system via PXE, selecting openfiler at the boot prompt, and then answering "NFS" and pointing it to the exported directory when it asked for the installation method. OpenFiler's Red Hat-like install (thanks to rPath) made installation very easy, and it installed very quickly over the LAN.

with a 2.6 kernel, very similar to Red Hat Linux. OpenFiler runs SSH by default, so you can just SSH to it and start poking around. The Web-based admin console is driven by Python and lighttpd. OpenFiler runs snmpd as well, so you can query it with SNMP. The HTTP/WebDAV engine appears to be Apache. It uses the standard Linux NFS server, has Samba to do the SMB/CIFS duty and leverages proftpd for its FTP server.

OpenFiler supports a wide range of physical block devices, like SATA, SAS, SCSI, IDE and FC disks. It also supports remote block devices, via the iSCSI, AoE (ATA over Ethernet) and FCoE (Fiber Channel over Ethernet) protocols. It supports the standard Linux software RAID as well.

One of the most interesting features of OpenFiler is the inclusion of the Distributed Replicated Block Device (DRBD) engine, as well as the Heartbeat HA cluster software. DRBD allows OpenFiler to replicate its block devices to another OpenFiler in either synchronous or asynchronous modes, so your backup OpenFiler could be in the next rack or in the next state. When combined with the Heartbeat HA software that allows two OpenFilers on the same LAN to use a Virtual IP address, you have a powerful, reliable, fault-tolerant data-storage cluster. In the event of a failure on the primary OpenFiler, the secondary will detect that across the private interconnect between the two units, step in, assume the virtual IP address and continue servicing requests.

Because OpenFiler uses Linux LVM, you easily can aggregate storage devices into a single pool and then slice that up as desired into whatever network share you want. Another benefit of using the Linux LVM is that point-in-time snapshots can be taken quickly and easily, allowing for consistent backups to be taken of the OpenFiler appliance.

Conclusion

OpenFiler is an easy-to-deploy and easy-to-use distribution that does one thing very well, and that's serve files to network clients. If you've got an older computer or laptop lying around, you can turn that system into a NAS appliance simply by installing OpenFiler and attaching a large USB disk. On the other end of the spectrum, OpenFiler is very well suited for installation on an enterprise-class server where it can act as a part of your corporate SAN. It's unfortunate that the developers elected to make the Administration Guide available to paying customers only, but the project needs to be funded by some means. If you've got a requirement for a file server or some form of networkable storage device, it's definitely worth checking out.■

Bill Childers is an IT Manager in Silicon Valley, where he lives with his wife and two children. He enjoys Linux far too much, and probably should get more sun from time to time. In his spare time, he does work with the Gilroy Garlic Festival, but he does not smell like garlic.

Resources

OpenFiler Home Page: **www.openfiler.com**

OpenFiler Architecture: **openfiler.com/products/openfiler-architecture**

OpenFiler Installation Documentation (Graphical): **www.openfiler.com/learn/how-to/graphical-installation**

OpenFiler Installation Documentation (Text): **www.openfiler.com/learn/how-to/graphical-installation**

OpenFiler 1.1 Admin Guide (downrev): **wwwold.openfiler.com/docs/manual**

DRBD (Distributed Replicated Block Device): **www.drbd.org**

Installing and Configuring OpenFiler with DRBD and Heartbeat: **www.howtoforge.com/installing-and-configuring-openfiler-with-drbd-and-heartbeat**

Unofficial OpenFiler HA Cluster Wiki: **wiki.hyber.dk/doku.php/openfiler_2.2_ha-cluster_guide**

AJAX

This month, our attention turns to one of the hottest areas for application development these days—AJAX.

KYLE RANKIN

BILL CHILDERS

Is AJAX the ideal way to build a cross-platform application, or is it just a rehash of the Java applets and CGI programs of yesteryear? Bill's opinion is Web 2.0-compliant, while Kyle's not very impressed and prefers native applications. Is AJAX the platform of the future or just a dancing bear? Read on for their take.

KYLE: So, Bill, what is so awesome about AJAX?

BILL: I dig using AJAX applications primarily because my computer becomes stateless. I don't have to worry about where that data is or installing some application—it's just there and ready for me to use.

KYLE: It seems like all those applications have already existed on the Web—they just were written in Java or some sort of CGI. I mean, I was chatting from a Web browser back in 1997.

BILL: Sure, there was a CGI chat, and I've seen Java applet chats too. But Web 2.0 is more than just chat applications, and besides, all those early apps had *horrid* usability issues.

KYLE: It just seems to me that AJAX suffers from the dancing-bear syndrome—people aren't impressed by how good the apps are, but that someone was able to get JavaScript to do it. I mean, ugly Java widgets aside, it seems like all these JavaScript apps existed years ago in other languages.

BILL: Where have you been, man? Sure, that was the case when the first AJAX apps came out that were really mind-blowing, like Google Maps. Even you have to admit that dragging the map around is a *huge* leap in usability.

KYLE: I remember the first time I saw Google Maps. I definitely was impressed that I could drag the map with my mouse and it moved, and zoomed, all within JavaScript. But, if that were a Java applet or a desktop program, no one would have cared nearly as much.

BILL: Now the applications have moved past the "gee whiz" factor and become full-fledged applications. Have you tried Google Calendar or Google Docs? Both of those are great examples. The Web interface that Zimbra uses for mail also is very good. It looks and feels a lot like most mail clients—to the point where people I've put on it have zero learning curve using it.

KYLE: That's exactly my point. What's impressive about those Web apps is that they almost act like a desktop application, yet if someone wrote the same thing as a desktop application, most people wouldn't be impressed. Okay, so I will confess. I do use Google Reader for RSS feeds, but honestly, the only thing it has over the Sage Firefox plugin is vi keybindings. I mean, Firefox already consumes enough memory as it is. The Web browser has become the new emacs: a single program that tries to do everything. It's the opposite of the "do one thing well" UNIX philosophy.

BILL: You use Google Reader! Blasphemy! That "do one thing well" UNIX philosophy is so dated, man. More and more and more programs are moving toward having multiple features and functions. It's what people *want* that drives that, not any overriding philosophy. People were talking about the browser *being* the OS back in 2005. AJAX applications help make that a reality. It's all about ubiquity—and the browser is the most ubiquitous part of any modern computer.

KYLE: That just sounds like the feature creep that we all used to complain about with Microsoft. Of course, Sun was talking about the network being the computer ages ago too, but then it needed to sell high-end servers. Is it really just the fact that Java widgets are pretty ugly that has caused everyone to rush to AJAX?

BILL: It's not feature creep...the application isn't part of the browser. If it were, then I'd agree with you. Java widgets are also somewhat fat, and there is the runtime compile issue, and the fact that despite Java's promise of "write once, run anywhere", that wasn't close to true until recently, and even now, it's not totally 100%.

KYLE: Well, at least Firefox has gotten good about restoring your sessions. If all of your apps are in the browser basket, you'd hope you wouldn't lose your work when that basket breaks.

BILL: If you're running programs within an X session and X barfs, you lose your work too. Regardless of what technology drives an application, it still runs within a container. If the container explodes, so does your app.

KYLE: I suppose I just disagree that the Web browser is the ideal container for all of my programs. Look at how much hacking it took so that these AJAX programs can maintain some sort of state when there is no Internet connection. With a desktop program, that's not even a concern.

BILL: That's true. Gears comes to mind to enable that, and that is kind of a hack. But honestly, how often are you without an Internet connection? I seem to remember you being *very* proud of configuring servers remotely from a Lake Tahoe mountaintop. If you have connectivity there, most likely you'll have it just about anywhere.

KYLE: Although these days it's much easier to have a connection anywhere you go, cell-phone tethering can be iffy at places, and I can't always drop a few bucks on a wireless connection at a coffee shop just to use a word processor (not that I'd use anything but vim anyway). Plus, what happens if you are in the middle of a program and your connection gets interrupted?

BILL: The Google stuff saves your work *very* frequently. I'd imagine you'd lose a sentence, maybe two, at most. It all depends on the application, doesn't it? If you lose the connection to your Google Calendar, it's not a big deal.

KYLE: My last word on the subject is just that I don't see much in AJAX that wasn't done under another Web technology years ago. It just seems like hype to me—everyone who is caught up in it thinks a program is instantly better when it runs from the Web and all the vowels are removed from its name. I think some things run better, and faster, on your own computer. After all, it seems a shame for all of the horsepower in Bill's planet-sized "laptop" to go to waste.

BILL: Yeah, AJAX is a newish Web technology (Google Maps came out with it in 2005—I hate to see what Kyle thinks is *old*). Despite that though, it's the first technology that actually enables developers to write compelling Web applications. Java applets were way off, and Java never quite got there. I'm rather shocked Kyle doesn't like it more, as his poor midget laptop probably could run the apps just fine. After all, if the iPhone can run an AJAX application, a "real computer" probably should be able to handle it too.■

Kyle Rankin is a Senior Systems Administrator in the San Francisco Bay Area and the author of a number of books, including *Knoppix Hacks* and *Ubuntu Hacks* for O'Reilly Media. He is currently the president of the North Bay Linux Users' Group.

Bill Childers is an IT Manager in Silicon Valley, where he lives with his wife and two children. He enjoys Linux far too much, and he probably should get more sun from time to time. In his spare time, he does work with the Gilroy Garlic Festival, but he does not smell like garlic.

Privacy Is Relative

Meaning, its context is relationship—or the absence of one.

DOC SEARLS

Years ago, I worked with PGP (Pretty Good Privacy) when it was a startup company and not what Phil Zimmerman created in the first place: a pretty good way to keep communications private. In the course of that work, I developed a belief that privacy was one of those topics that was too important to ignore, yet too complex for most people to understand, especially if it involved technology more complex than a key and a hole. So I've mostly avoided the topic, leaving the worrying up to people who are required to wrestle with it—meaning, developers.

But now, I'm running a development project, and not a day goes by that privacy doesn't come up—or worse, require consequential thinking about nitty-gritties: code, protocols, policies and (worst of all) legal stuff. So I've been trying to think in new ways about privacy—what it means and how to put that meaning to work.

Let's start with celebrities. These creatures can play a helpful role in studies of privacy, because they have less of it than the rest of us. Celebrity is a kind of albinism. It robs its victims of the pigment we call anonymity, even as they are dressed in fame. So they stand out. Worse, they attract the attention of paparazzi, whose purpose in life is to maximize celebrity exposure.

Mass media (the natural environment of celebrity) reduce and confine the degree to which celebrities can enjoy simple one-to-one, or one-to-any, relationships. So celebrities hide. Or give up. Or both.

Scott McNealy famously said, "You have no privacy. Get over it." Asked by a gaggle of *San Francisco Chronicle* reporters to expand on that, he replied, "The point I was making was someone already has your medical records. Someone has my dental records. Someone has my financial records. Someone knows just about everything about me. Gang, do you want to refute my statement? Visa knows what you bought. You have no privacy. Get over it. That's what I said."

For years I thought, "Well, that's true for him." Because he's a celebrity. But lately, I've thought more about the rest of what he said, which was about data. The fact is, your medical, financial and dental records are not yours. They might be about you, but they don't belong to you. They belong to your credit-card company, your broker, your dentist.

We go to those professionals because we can't or won't perform their work by ourselves. So, because they're the ones producing data about us, it only makes sense for the data to be "theirs"—at least in the locational sense. After that, the distinction between control and possession comes up only when somebody else needs the data. If that's you, all you need to do in most cases is authenticate yourself. Then you can have it.

In the physical world, that's fairly easy. We just show up looking like ourselves. If we have a familiar working relationship with our dentists, bankers or brokers, they won't bother asking for our drivers' licenses. They'll just shake our hands, tell us to have a seat and ask us how we're doing.

This illustrates how there are essentially two forms of privacy. One is the kind where you hide out. You minimize exposure by confining it to yourself. The other is where you trust somebody with your information.

In order to trust somebody, you need a relationship with them. You're their spouse, friend, client or patient.

This isn't so easy if you're just a customer, or worse, a "consumer". There the obligation is minimized, usually through call centers and other customer-avoidance mechanisms that get only worse as technology improves. Today, the call center wants to scrape you off onto a Web site or a chat system.

Minimizing human contact isolates your private information inside machines that have little interest in relating to you as a human being or in putting you in contact with a human being inside the company. Hence, your data is indeed safe—from you. It's also safe from the assumption that this data might in any way also belong to you—meaning, under your control. It's still private, but only on the company's terms. Not on yours.

This mess can't be fixed just by humanizing call centers. It can be fixed only by humanizing companies. This has to be done from both inside and out.

Recent changes in the sounds coming from the CRM community are highly encouraging. So is the growth of free and open-source CRM systems and the interest of CRM giants such as Oracle in VRM (vendor relationship management), which is the development movement I'm involved in.

Paul Trevithick, the main developer behind Higgins (**www.eclipse.org/higgins**), makes an interesting point: both the Net and the Web were born without the concept of an individual. There are endpoints on the Net and files on the Web—and the presumption that somebody will do browsing or viewing. But here is no instantiation of the individual himself or herself, except inside company silos.

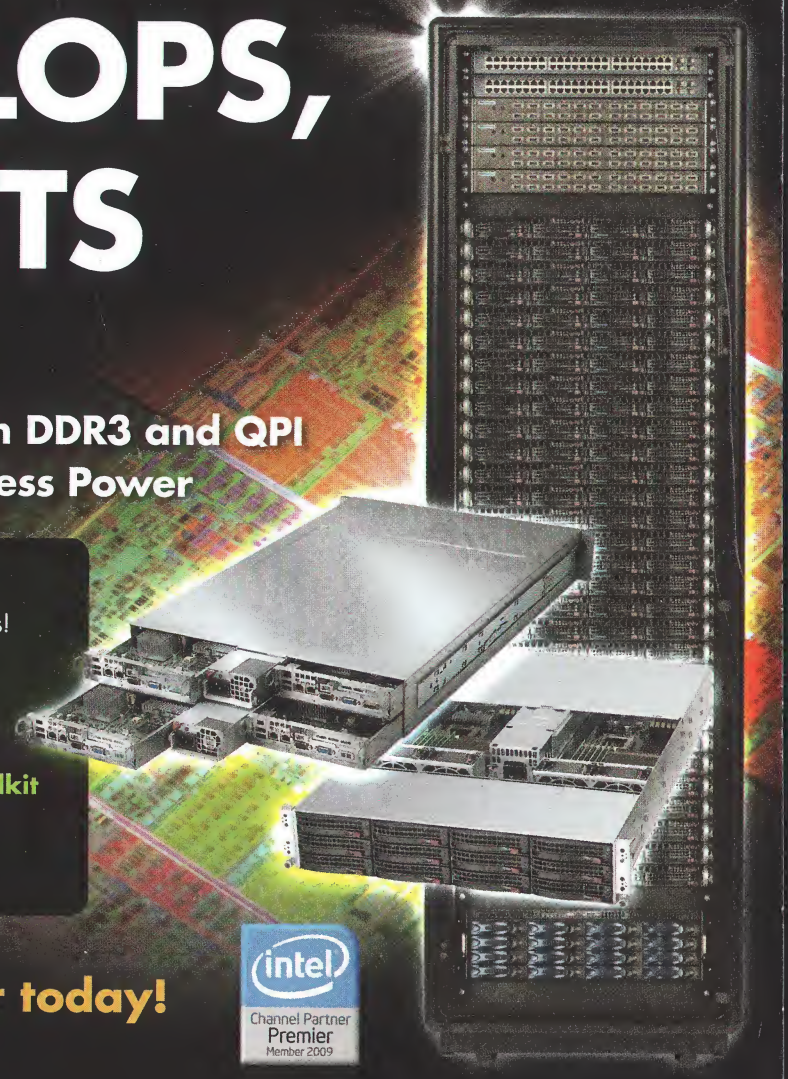
Keith Hopper says, "The customer should be his own silo." Building those won't be easy, but it will be necessary if we want privacy that's more than pretty good. Those silos will have two effects. One is to contain our data and put it under our control. The other is to position us as an equal: a free and independent entity rather than a captive and dependent one.■

Doc Searls is Senior Editor of *Linux Journal*. He is also a fellow with the Berkman Center for Internet and Society at Harvard University and the Center for Information Technology and Society at UC Santa Barbara.

More GFLOPS, Less WATTS

Intel® Nehalem is here!

Higher Memory Bandwidth with DDR3 and QPI
Clusters and Servers Consume Less Power



Four Servers in a 2U Chassis with all Hot-Swap:
- 1200 Watt 1+1 supply, 12 Drives, and Server Modules!

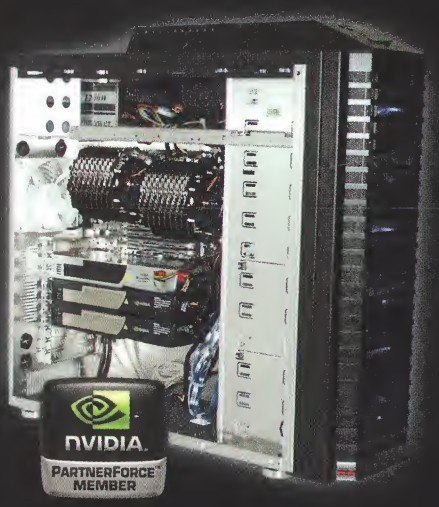
FasTree™ ConnectX® QDR and DDR InfiniBand Switches and HCAs

Intel Professional Compiler Suite and Cluster Toolkit
- Version 11 with Nehalem Enhancements
- Academic Pricing Available

intel Channel Partner Premier Member 2009

Configure your next Cluster today!

www.microway.com/quickquote



GPU Computing

WhisperStation™

With 1 to 4 Tesla GPUs

Tesla C1060 GPU Performance:
- 1 TFLOPS per GPU
- 4 GB DDR3 per GPU
- 102 GB/Sec Bandwidth
- CUDA SDK

Run MATLAB® on Tesla with "Jacket"

Clusters With Tesla™

S1070 - 4 GPU Servers
- 36 GPUs + 36 CPUs + 24 TB in 24U
- 40 Gbps FasTree™ InfiniBand
- InfiniScope™ Network Monitoring

FREE 15-day trial available at **microway.com**

Microway®
Technology you can count on℠

508-746-7341
microway.com

GSA Schedule Contract Number: GS-35F-0431N